Oracle® Linux

Security Guide for Release 6

E36387-10
November 2014

Oracle® Linux: Security Guide for Release 6

Abstract

This manual provides security guidelines for the Oracle Linux 6 operating system.

Document generated on: 2014-11-25 (revision: 2420)

Table of Contents

Preface v
1 Oracle Linux Security Overview 1
    1.1 Basic Security Considerations 1
        1.1.1 Keep Software up to Date 1
        1.1.2 Restrict Network Access to Critical Services 1
        1.1.3 Follow the Principle of Least Privilege 1
        1.1.4 Monitor System Activity 2
        1.1.5 Keep up to Date on the Latest Security Information 2
    1.2 The Oracle Linux Security Model 2
    1.3 Overview of Oracle Linux Security 2
    1.4 Understanding the Oracle Linux Environment 3
    1.5 Recommended Deployment Configurations 3
    1.6 Component Security 4
    1.7 References 4
2 Secure Installation and Configuration 7
    2.1 Pre-Installation Tasks 7
    2.2 Installing Oracle Linux 7
        2.2.1 Shadow Passwords and Hashing Algorithms 8
        2.2.2 Strong Passwords 8
        2.2.3 Separate Disk Partitions 8
        2.2.4 Encrypted Disk Partitions 8
        2.2.5 Software Selection 9
        2.2.6 Network Time Service 9
    2.3 Post-Installation Tasks 9
3 Implementing Oracle Linux Security 11
    3.1 Configuring and Using Data Encryption 11
    3.2 Configuring a GRUB Password 12
    3.3 Configuring and Using Certificate Management 13
        3.3.1 About the openssl Command 14
        3.3.2 About the keytool Command 16
    3.4 Configuring and Using Authentication 17
        3.4.1 About Local Oracle Linux Authentication 18
        3.4.2 About IPA 18
        3.4.3 About LDAP Authentication 19
        3.4.4 About NIS Authentication 20
        3.4.5 About Winbind Authentication 20
        3.4.6 About Kerberos Authentication 21
    3.5 Configuring and Using Pluggable Authentication Modules 22
    3.6 Configuring and Using Access Control Lists 24
    3.7 Configuring and Using SELinux 24
        3.7.1 About SELinux Administration 26
        3.7.2 About SELinux Modes 27
        3.7.3 Setting SELinux Modes 27
        3.7.4 About SELinux Policies 28
        3.7.5 About SELinux Context 29
        3.7.6 About SELinux Users 32
    3.8 Configuring and Using Auditing 33
    3.9 Configuring and Using System Logging 34
    3.10 Configuring and Using Process Accounting 37
    3.11 Configuring and Using Software Management 37
        3.11.1 Configuring Update and Patch Management 39

3.11.2 Installing and Using the Yum Security Plugin 39
3.12 Configuring Access to Network Services 42
3.12.1 Configuring and Using Packet-filtering Firewalls 43
3.12.2 Configuring and Using TCP Wrappers 46
3.13 Configuring and Using Chroot Jails 47
3.13.1 Running DNS and FTP Services in a Chroot Jail 47
3.13.2 Creating a Chroot Jail 47
3.13.3 Using a Chroot Jail 48
3.14 Configuring and Using Linux Containers 48
3.15 Configuring and Using Kernel Security Mechanisms 49
3.15.1 Address Space Layout Randomization 49
3.15.2 Data Execution Prevention 49
3.15.3 Position Independent Executables 50
4 Security Considerations for Developers 51
4.1 Design Principles for Secure Coding 51
4.2 General Guidelines for Secure Coding 52
4.3 General Guidelines for Network Programs 53
5 Secure Deployment Checklist 55
5.1 Minimizing the Software Footprint 55
5.2 Configuring System Logging 56
5.3 Disabling Core Dumps 57
5.4 Minimizing Active Services 57
5.5 Locking Down Network Services 60
5.6 Configuring a Packet-filtering Firewall 61
5.7 Configuring TCP Wrappers 61
5.8 Configuring Kernel Parameters 61
5.9 Restricting Access to SSH Connections 62
5.10 Configuring File System Mounts, File Permissions, and File Ownerships 62
5.11 Checking User Accounts and Privileges 64
6 Enabling FIPS Mode for OpenSSL 67
6.1 About the FIPS 140-2 OpenSSL Library Modules 67
6.2 Configuring a System for FIPS Mode 67
6.3 Using the FIPS-Compliant OpenSSL Library 69

Preface

The *Oracle Linux Security Guide* provides security guidelines for the Oracle Linux 6 operating system. The guide presents steps that you can take to harden an Oracle Linux system and the features that you can use to protect your data and applications. You can tailor the recommendations in the guide to suit your site security policy.

Audience

This document is intended for administrators who analyze security requirements, implement site security policy, install and configure the Oracle Linux operating system, and maintain system and network security. It is assumed that readers have a general knowledge of Linux administration, a good foundation in software security, and knowledge of your organization's site security policy.

Document Organization

The document is organized as follows:

- Chapter 1, *Oracle Linux Security Overview* provides an overview of Oracle Linux security.
- Chapter 2, *Secure Installation and Configuration* outlines the planning process for a secure installation and describes how the choices that you make during installation affect system security.
- Chapter 3, *Implementing Oracle Linux Security* describes the various ways in which you can configure the security of an Oracle Linux system.
- Chapter 4, *Security Considerations for Developers* provides information for developers about how to create secure applications for Oracle Linux, and how to extend Oracle Linux to access external systems without compromising security.
- Chapter 5, *Secure Deployment Checklist* provide guidelines that help secure your Oracle Linux system.
- Chapter 6, *Enabling FIPS Mode for OpenSSL* describes how to make an Oracle Linux 6 system compliant with Federal Information Processing Standard (FIPS) Publication 140-2.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

The documentation for this product is available at:

http://www.oracle.com/technetwork/server-storage/linux/documentation/index.html.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Chapter 1 Oracle Linux Security Overview

Table of Contents

1.1 Basic Security Considerations 1
1.1.1 Keep Software up to Date 1
1.1.2 Restrict Network Access to Critical Services 1
1.1.3 Follow the Principle of Least Privilege 1
1.1.4 Monitor System Activity 2
1.1.5 Keep up to Date on the Latest Security Information 2
1.2 The Oracle Linux Security Model 2
1.3 Overview of Oracle Linux Security 2
1.4 Understanding the Oracle Linux Environment 3
1.5 Recommended Deployment Configurations 3
1.6 Component Security 4
1.7 References 4

This chapter provides an overview of Oracle Linux security.

1.1 Basic Security Considerations

The following sections list the fundamental principles for using Oracle Linux securely.

1.1.1 Keep Software up to Date

One of the principles of good security practice is to keep all software versions and patches up to date. Throughout this document, we assume a maintenance level of Oracle Linux Release 6 or later.

For more information, see Section 3.11, “Configuring and Using Software Management”

1.1.2 Restrict Network Access to Critical Services

Keep both middle-tier applications and databases behind a firewall. In addition, place a firewall between middle-tier applications and databases if these are hosted on separate servers. The firewalls provide assurance that access to these systems is restricted to a known network route, which can be monitored and restricted, if necessary. As an alternative, a firewall router substitutes for multiple, independent firewalls.

If firewalls cannot be used, restrict access based upon IP address. Restricting database access by IP address often causes application client/server programs to fail for DHCP clients. To resolve this, consider using static IP addresses, a software/hardware VPN or Windows Terminal Services or its equivalent.

For more information, see Section 3.12, “Configuring Access to Network Services”.

1.1.3 Follow the Principle of Least Privilege

The principle of least privilege states that users should be given the least amount of privilege to perform their jobs. Over ambitious granting of responsibilities, roles, grants, and so on, especially early on in an organization’s life cycle when people are few and work needs to be done quickly, often leaves a system wide open for abuse. User privileges should be reviewed periodically to determine relevance to current job responsibilities.

For more information, see Section 5.11, “Checking User Accounts and Privileges”.

1.1.4 Monitor System Activity

System security stands on three legs: good security protocols, proper system configuration, and system monitoring. Auditing and reviewing audit records address the third requirement. Each component within a system has some degree of monitoring capability. Follow audit advice in this document and regularly monitor audit records.

For more information, see Section 3.8, “Configuring and Using Auditing”.

1.1.5 Keep up to Date on the Latest Security Information

Oracle continually improves its software and documentation. Check regularly on the Oracle Technology Network at http://www.oracle.com/technetwork/server-storage/linux for revisions. For information about common vulnerabilities and exposures (CVE) and errata that are available on the Unbreakable Linux Network, see http://linux.oracle.com/cve and http://linux.oracle.com/errata.

1.2 The Oracle Linux Security Model

Oracle Linux provides a complete security stack, from network firewall control to access control security policies, and is designed to be secure by default.

Traditional Linux security is based on a Discretionary Access Control (DAC) policy, which provides minimal protection from broken software or from malware that is running as a normal user or as `root`. The SELinux enhancement to the Linux kernel implements the Mandatory Access Control (MAC) policy, which allows you to define a security policy that provides granular permissions for all users, programs, processes, files, and devices. The kernel's access control decisions are based on all the security relevant information available, and not solely on the authenticated user identity. By default, SELinux is enabled when you install an Oracle Linux system.

For more information, see Section 3.7, “Configuring and Using SELinux”.

1.3 Overview of Oracle Linux Security

Oracle Linux has evolved into a secure enterprise-class operating system that can provide the performance, data integrity, and application uptime necessary for business-critical production environments.

Thousands of production systems at Oracle run Oracle Linux and numerous internal developers use it as their development platform. Oracle Linux is also at the heart of several Oracle engineered systems, including the Oracle Exadata Database Machine, Oracle Exalytics In-Memory Machine, Oracle Exalogic Elastic Cloud, and Oracle Database Appliance.

Oracle On Demand services, which deliver software as a service (SaaS) at a customer's site, via an Oracle data center, or at a partner site, use Oracle Linux at the foundation of their solution architectures. Backed by Oracle support, these mission-critical systems and deployments depend fundamentally on the built-in security and reliability features of the Oracle Linux operating system.

Released under an open-source license, Oracle Linux includes the Unbreakable Enterprise Kernel that provides the latest Linux innovations while offering tested performance and stability. Oracle has been a key participant in the Linux community, contributing code enhancements such as Oracle Cluster File System and the Btrfs file system. From a security perspective, having roots in open source is a significant advantage. The Linux community, which includes many experienced developers and security experts, reviews posted Linux code extensively prior to its testing and release. The open-source Linux community has supplied many security improvements over time, including access control lists (ACLs), cryptographic libraries, and trusted utilities.

1.4 Understanding the Oracle Linux Environment

To better understand your security needs, ask yourself the following questions:

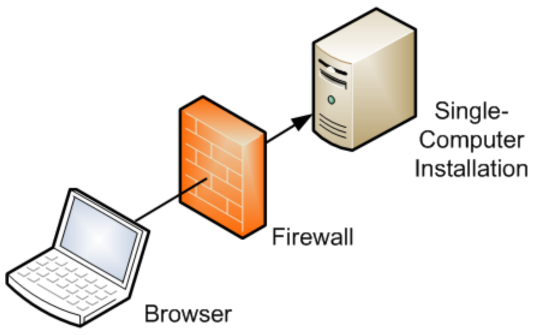
Which resources am I protecting?	Many resources in the production environment can be protected, including information in databases accessed by WebLogic Server and the availability, performance, applications, and the integrity of the Web site. Consider the resources you want to protect when deciding the level of security you must provide.
From whom am I protecting the resources?	For most Web sites, resources must be protected from everyone on the Internet. But should the Web site be protected from the employees on the intranet in your enterprise? Should your employees have access to all resources within the WebLogic Server environment? Should the system administrators have access to all WebLogic resources? Should the system administrators be able to access all data? You might consider giving access to highly confidential data or strategic resources to only a few well trusted system administrators. Perhaps it would be best to allow no system administrators access to the data or resources.
What will happen if the protections on strategic resources fail?	In some cases, a fault in your security scheme is easily detected and considered nothing more than an inconvenience. In other cases, a fault might cause great damage to companies or individual clients that use the Web site. Understanding the security ramifications of each resource will help you protect it properly.

1.5 Recommended Deployment Configurations

This section describes recommended architectures for deploying Oracle products with secure Internet access.

Figure 1.1, “Simple Firewall Deployment Configuration” shows a simple deployment architecture.

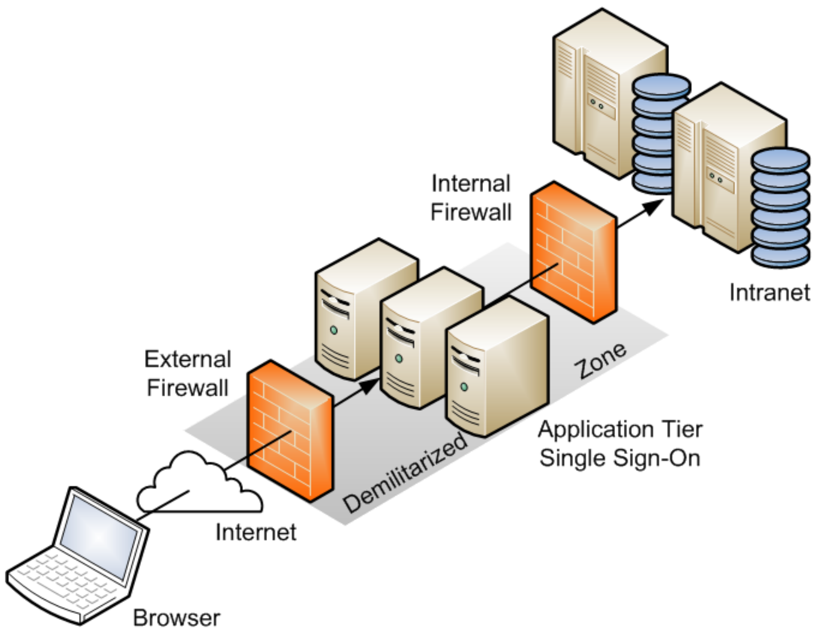
Figure 1.1 Simple Firewall Deployment Configuration



This single-computer deployment may be cost effective for small organizations. However, it cannot provide high availability because all components are stored on the same computer.

Figure 1.2, “DMZ Deployment Configuration” shows the recommended configuration, which uses the well-known and generally accepted Internet-Firewall-DMZ-Firewall-Intranet architecture.

Figure 1.2 DMZ Deployment Configuration



A *demilitarized zone* (DMZ) refers to a server that is isolated by firewalls from both the Internet and the intranet, and which acts a buffer between them. The firewalls that separate DMZ zones provide two essential functions:

- Block any traffic types that are not permitted.
- Provide intrusion containment in the event that successful intrusions take over processes or processors.

1.6 Component Security

Each application software component usually has its own security considerations that you should take into account independently of those that apply to the operating system. Refer to the security guidelines for each component to determine how best to configure it for the requirements of security at your site.

1.7 References

For more information on the security topics covered in this guide, see the following references:

NIST *Checklist Details for DoD Consensus Security Configuration Checklist for Red Hat Enterprise Linux 5 2.0*: http://web.nvd.nist.gov/view/ncp/repository/checklistDetail?id=294.

NIST *SCAP: Guide To The Secure Configuration of Red Hat Enterprise Linux 5*: http://nvd.nist.gov/scap/content/stylesheet/scap-rhel5-document.htm.

NSA *Guide to the Secure Configuration of Red Hat Enterprise Linux 5*: http://www.nsa.gov/ia/_files/os/redhat/NSA_RHEL_5_GUIDE_v4.2.pdf.

NSA *Hardening Tips For Default Installation of Red Hat Enterprise Linux 5*: http://www.nsa.gov/ia/_files/factsheets/rhel5-pamphlet-i731.pdf.

NSA *Meeting Critical Security Objectives with Security-Enhanced Linux* by Peter A. Loscocco and Stephen D. Smalley: http://www.nsa.gov/research/_files/selinux/papers/ottawa01/index.shtml.

NSA *SELinux Frequently Asked Questions (FAQ)*: http://www.nsa.gov/research/selinux/faqs.shtml.

Oracle *Tips for Hardening an Oracle Linux Server* by Lenz Grimmer and James Morris: http://www.oracle.com/technetwork/articles/servers-storage-admin/tips-harden-oracle-linux-1695888.html.

Oracle *Tips for Securing an Oracle Linux Environment* by Ginny Henningsen, James Morris, and Lenz Grimmer: http://www.oracle.com/technetwork/articles/servers-storage-admin/secure-linux-env-1841089.html.

SELinux Project Wiki: http://selinuxproject.org/page/Main_Page.

Note

The most up-to-date advice that is available from some of the sources applies to Oracle Linux 5. However, most of the advice is also applicable to Oracle Linux 6.

Chapter 2 Secure Installation and Configuration

Table of Contents

2.1 Pre-Installation Tasks 7
2.2 Installing Oracle Linux 7
2.2.1 Shadow Passwords and Hashing Algorithms 8
2.2.2 Strong Passwords 8
2.2.3 Separate Disk Partitions 8
2.2.4 Encrypted Disk Partitions 8
2.2.5 Software Selection 9
2.2.6 Network Time Service 9
2.3 Post-Installation Tasks 9

This chapter outlines the planning process for a secure installation and describes how the choices that you make during installation affect system security.

2.1 Pre-Installation Tasks

An important consideration is the security of the physical system on which you will install Oracle Linux. If possible, keep server systems in a locked data center and limit access to authorized personnel. Such personnel should also receive appropriate administrative training as human error is often the cause of a security breach. For more information about the available Oracle Linux coursework and certification options, see http://education.oracle.com.

Aside from the risks of theft and data compromise, physical security is critical because it prevents an unauthorized user from possibly modifying the system BIOS, altering the boot device, and booting from an alternate medium. If a system is not kept in a locked data center, consider password-protecting the BIOS. Consult the system manufacturer's documentation for information on setting a BIOS password. Edit the BIOS settings to disable booting from the CD-ROM drive, floppy disk drive, USB ports, and other external devices. In addition, you can configure disk encryption during installation, or password-protect the GRUB boot loader after installation.

Note

Setting a BIOS, encrypted disk, or boot-loader password requires you to enter the password whenever you reboot the system. Only disk encryption can prevent access to the data on disk when an attacker uses techniques such as resetting the BIOS, accessing the disk by booting an operating system from a memory stick, or simply removing the hard drive to read its contents on another system.

2.2 Installing Oracle Linux

When you install Oracle Linux, you can reduce the attack surface by installing only the software packages that are required for operation. Software packages are a potential source of `setuid` programs, network services, and libraries that an attacker can potentially use to gain access illegitimately and compromise a system.

You can use a pretested Kickstart profile to provide consistent and precise control over what is installed. Automated installation using a Kickstart profile reduces both security risk and administrative effort.

Alternatively, you can use Oracle Enterprise Manager Ops Center, which supports the import of OS images and explicit provisioning profiles. For more information, refer to the Oracle Enterprise Manager Ops Center documentation.

2.2.1 Shadow Passwords and Hashing Algorithms

By default, an Oracle Linux system is configured to use password hashes that are stored in the `/etc/shadow` file rather than in the world-readable `/etc/passwd` file. If shadow passwords were not used, an attacker is much more likely to be able to discover a password by applying cracking software to the hashes. Similarly, using a password-hashing algorithm that is weaker than SHA-512 would make it much easier to find likely candidates that match a hash value.

2.2.2 Strong Passwords

During installation, you are prompted to enter passwords for `root` and one additional user, if you choose the user to be authenticated locally rather than over the network. The passwords that you enter should be strong in that they should be extremely difficult to deduce by guesswork or by other means, such as automated FTP or SSH logins. By default, the installation process rejects null passwords and warns about weak passwords, but it does not enforce strong passwords. It is your responsibility to ensure that passwords are sufficiently strong.

Some general guidelines for creating a strong password are:

- Make the password at least eight characters long.
- Use a mixture of lower and upper case letters, numbers, and other characters.
- Do not include whole words from English, LEET speak, or any other language or technology, even if you spell the words in reverse order. Password cracking programs are more sophisticated that one might naively assume.
- Do not include personal information such as names, dates, addresses, email addresses, or telephone numbers.
- Do not use well-known acronyms, abbreviations, or character sequences such as QWERTY.
- Do not use a password that is the same as or very similar to a password that you used previously on the system.
- Use a password for `root` that is different from the password for any other user.

2.2.3 Separate Disk Partitions

The National Security Agency (NSA) recommendations state that you should set up user-writable file systems such as `/home`, `/tmp`, and `/var/tmp` on partitions that are separate from `/`. In addition, `/boot` must be a dedicated file system if you encrypt the `root` file system.

For more information, see http://www.nsa.gov/ia/_files/factsheets/rhel5-pamphlet-i731.pdf.

2.2.4 Encrypted Disk Partitions

When choosing a disk layout, you have the option of encrypting disk partitions with the Linux Unified Key Setup (LUKS) format. As for any other password, ensure that you enter a strong passphrase if you choose to encrypt any partitions.

Note

The `/boot` file system cannot be encrypted.

2.2.5 Software Selection

If you choose to customize the software to be installed on a system, you can select or deselect packages from the default set. For example, the basic server configuration does not install the Gnome and KDE desktop software and the X Windows System packages from the **Desktops** section. Additional packages that you might want to install on a server system are available under the **Servers**, **Web Services**, **Databases**, and other section headings.

2.2.6 Network Time Service

If you select to synchronize the data and time over the network, the system is configured as an NTP client that uses the `[012].rhel.pool.ntp.org` public servers by default. If your systems rely on Kerberos authentication, which requires close synchronization of the clocks on each participating system, you might prefer to configure your systems to use a local NTP server instead.

2.3 Post-Installation Tasks

For information about the way that you can configure the security of an Oracle Linux system, see Chapter 3, *Implementing Oracle Linux Security*.

For guidelines about hardening an Oracle Linux system, see Chapter 5, *Secure Deployment Checklist*.

Chapter 3 Implementing Oracle Linux Security

Table of Contents

3.1 Configuring and Using Data Encryption 11
3.2 Configuring a GRUB Password 12
3.3 Configuring and Using Certificate Management 13
3.3.1 About the openssl Command 14
3.3.2 About the keytool Command 16
3.4 Configuring and Using Authentication 17
3.4.1 About Local Oracle Linux Authentication 18
3.4.2 About IPA 18
3.4.3 About LDAP Authentication 19
3.4.4 About NIS Authentication 20
3.4.5 About Winbind Authentication 20
3.4.6 About Kerberos Authentication 21
3.5 Configuring and Using Pluggable Authentication Modules 22
3.6 Configuring and Using Access Control Lists 24
3.7 Configuring and Using SELinux 24
3.7.1 About SELinux Administration 26
3.7.2 About SELinux Modes 27
3.7.3 Setting SELinux Modes 27
3.7.4 About SELinux Policies 28
3.7.5 About SELinux Context 29
3.7.6 About SELinux Users 32
3.8 Configuring and Using Auditing 33
3.9 Configuring and Using System Logging 34
3.10 Configuring and Using Process Accounting 37
3.11 Configuring and Using Software Management 37
3.11.1 Configuring Update and Patch Management 39
3.11.2 Installing and Using the Yum Security Plugin 39
3.12 Configuring Access to Network Services 42
3.12.1 Configuring and Using Packet-filtering Firewalls 43
3.12.2 Configuring and Using TCP Wrappers 46
3.13 Configuring and Using Chroot Jails 47
3.13.1 Running DNS and FTP Services in a Chroot Jail 47
3.13.2 Creating a Chroot Jail 47
3.13.3 Using a Chroot Jail 48
3.14 Configuring and Using Linux Containers 48
3.15 Configuring and Using Kernel Security Mechanisms 49
3.15.1 Address Space Layout Randomization 49
3.15.2 Data Execution Prevention 49
3.15.3 Position Independent Executables 50

This chapter describes the various ways in which you can configure the security of an Oracle Linux system.

3.1 Configuring and Using Data Encryption

You can use data encryption to protect data that is stored or that is being transmitted. Data on storage devices and media can be at risk of theft or device loss. Data being transmitted over local area networks and the Internet can be intercepted or altered. In addition, data encryption to protect privacy and personal data is increasingly being made a mandatory requirement of corporate security policy and by governmental regulations (for example, HIPAA, GLBA, SOX, and PCI DSS).

Oracle Linux systems provide several strategies for protecting data:

- When installing systems and application software, only accept RPM packages that have been digitally signed. To ensure that downloaded software packages are signed, set `gpgcheck=1` in the repository configuration file and import the GPG key provided by the software supplier. You can also install RPMs using the Secure Sockets Layer (SSL) protocol, which uses encryption to protect the communications channel.

- To protect against data theft, consider using full-disk encryption, especially on laptops, external hard drives, or removable devices such as USB memory sticks. Oracle Linux supports block device encryption using `dm-crypt` and the Linux Unified Key Setup (LUKS) format. The `cryptsetup` administration command is available in the `cryptsetup-luks` package. These technologies encrypt device partitions so that the data is inaccessible when a system is turned off. When the system boots and you supply the appropriate passphrase, the device is decrypted and its data is accessible. For more infomation, see the `cryptsetup(8)` manual page.

- An alternative approach for protecting data on a device is to use the eCryptfs utilities to encrypt a file system. The eCryptfs utilities are available in the `ecryptfs-utils` package. Unlike `dm-crypt`, which encrypts block devices, eCryptfs encrypts data at the file-system level, and you can also use it to protect individual files and directories. For more information, see the `ecryptfs(7)`, `ecryptfs-setup-private(1)`, `ecryptfs-mount-private(1)`, and `ecryptfs-umount-private(1)` manual pages.

- Oracle Linux uses encryption to support Virtual Private Networks (VPN), Secure Shell (`ssh`), and password protection. By default, Oracle Linux uses a strong password hashing algorithm (SHA-512) and stores hashed passwords in the `/etc/shadow` file.

- Oracle Linux takes advantage of hardware-accelerated encryption on Intel CPUs that support the Advanced Encryption Standard New Instructions (AES-NI) instruction set, which speeds up the execution of AES algorithms as well as SHA-1 and RC4 algorithms on x86 and x86_64 architectures.

3.2 Configuring a GRUB Password

If a system is not kept in a locked data center, and as an alternative to using any password protection mechanism built into the BIOS, you can add a degree of protection to the system by requiring a valid password be provided to the GRUB boot loader.

Note

Password protecting GRUB access prevents unauthorized users from entering single user mode and changing settings at boot time. It does not prevent someone from accessing data on the hard drive by booting into an operating system from a memory stick, or physically removing the drive to read its contents on another system.

To configure a GRUB password:

1. Use the following command to generate the MD5 hash of your password:

```
# /sbin/grub-md5-crypt
Password: clydenw
Retype password: clydenw
$1$qhqh.1$7MQxS6GHg4IlOFMdnDx9S.
```

2. Edit `/boot/grub/grub.conf`, and add a `password` entry below the `timeout` entry near the top of the file, for example:

   ```
   timeout=5
   password --md5 pwhash
   ```

   where *`pwhash`* is the hash value that `grub-md5-crypt` returned.

3. If GRUB has been configured to boot multiple operating systems on the same machine, add a `lock` entry to after the `title` entry for each operating system, for example:

   ```
   title Windows
   lock
   ```

When you reboot the machine, you must press `P` and enter the password before you can access the GRUB command interface.

For more information, use the `info grub` command to access the GRUB manual.

3.3 Configuring and Using Certificate Management

Public-key cryptography allows secure communication on an insecure public network and verification of the identity of the entity at the other end of a network connection. Public-key cryptography is based on establishing pairs of secret and public keys. Either key can be used to encrypt some data, and the other key can then be used to decrypt that data. You cannot use just one of the keys to perform both operations on the same data. Because of the asymmetric nature of the key operations, you can distribute the public key without fear of compromising security. Possession of the private key is required to be able to read messages that are encrypted with the public key.

However, if you receive a public key, this in itself does not establish the identity of the sender. A public-key infrastructure implements digital certificates that allow public keys to be distributed in a hierarchy of trusted relationships. A Certification Authority (CA) acts as a trusted third party that can issue signed certificates on behalf of another entity on a network. The CA uses its own private key to encrypt the certificate, which contains the entity's public key together with other information about the entity (*subject*), the CA (*issuer*), the period of validity of the certificate, and the cryptographic algorithms used. Assuming that you trust the CA, you can also trust the entity's public key stored in the certificate. Decrypting the certificate with the CA's public key yields the entity's public key, and this key can be used to establish a secure communications channel.

For the Internet, there are many public top-level (*root*) CAs and there are also many intermediate CAs that are trusted by a root CA to issue certificates on behalf of entities. An intermediate CA usually returns a certificate chain, where each certificate in the chain authenticates the public key of the signer of the previous certificate in the chain up to and including a root CA. The secure communication channels that are required for website security usually use the Transport Layer Security (TLS) or Secure Sockets Layer (SSL) cryptographic protocols. Because most financial transactions on the Internet rely on TLS and SSL, a more limited number of CAs are permitted to issue TLS/SSL certificates that web browsers trust, and these CAs are regularly audited for security.

OpenSSL is an open-source implementation of the TLS and SSL protocols. If a hierarchy of trust is confined to your organization's intranet, you can use OpenSSL to generate a root certificate and set up a CA for that domain. However, unless you install this self-signed root certificate on each system in your organization, browsers, LDAP or IPA authentication, and other software that use certificates will prompt the user about the potentially untrusted relationship. If you use certificates for your domain that are validated by a root or intermediate-level CA, you do not need to distribute a root certificate as the appropriate certificate should already be present on each system.

Typically, TLS/SSL certificates expire after one year. Other certificates, including root certificates that are distributed with web browsers, and which are issued by root and intermediate CAs usually expire after a period from five to 10 years. To avoid applications displaying warnings about out-of-date certificates, you should plan to replace TLS/SSL certificates before they expire. For root certificates, this is not usually a problem as you would typically update the software before the certificate would expire.

If you request a signed certificate from a CA for which a root certificate or certificate chain that authenticates the CA's public key does not already exist on your system, obtain a trusted root certificate from the CA. To avoid a potential man-in-the-middle attack, verify the authenticity of the root certificate before importing it. Check that the certificate's fingerprint matches the fingerprint that the CA publishes.

The `openssl` command allows you to generate self-signed certificates that web browsers can use. You can also use the `keytool` command to generate self-signed certificates, but this command's primary purpose is to install and manage JSSE (Java Secure Socket Extension) digital certificates for use with Java applications.

Note

For production environments, you should obtain external CA-signed certificates, which can be revoked if the private key is compromised. Self-signed certificates cannot be revoked, and should only be used when developing, testing, or demonstrating software.

For more information about using TLS/SSL and certificates with the Apache HTTP server, see the Apache documentation at http://httpd.apache.org/docs.

3.3.1 About the openssl Command

The `openssl` command, which is included in the `openssl` package, allows you to perform various cryptography functions from the OpenSSL library including:

- Creating and managing pairs of private and public keys.
- Performing public key cryptographic operations.
- Creating self-signed certificates.
- Creating certificate signing requests (CSRs).
- Creating certificate revocation lists (CRLs).
- Converting certificate files between various formats.
- Calculating message digests.
- Encrypting and decrypting files.
- Testing both client-side and server-side TLS/SSL with HTTP and SMTP servers.
- Verifying, encrypting and signing S/MIME email.
- Generating and testing prime numbers, and generating pseudo-random data.

The following are some sample `openssl` commands.

Create a self-signed X.509 certificate that is valid for 365 days, writing the unencrypted private key to `prikey.pem` and the certificate to `cert.pem`.

```
# openssl req -x509 -nodes -days 365 -subj '/C=US/ST=Ca/L=Sunnydale/CN=www.unserdom.com' \
  -newkey rsa:1024 -keyout prikey.pem -out cert.pem
```

Test a self-signed certificate by launching a server that listens on port 443.

```
# openssl s_server -accept 443 -cert cert.pem -key prikey.pem -www
```

Test the client side of a connection. This command returns information about the connection including the certificate, and allows you to directly input HTTP commands.

```
# openssl s_client -connect server:443 -CAfile cert.pem
```

Convert a root certificate to a form that can be published on a web site for downloading by a browser.

```
# openssl x509 -in cert.pem -out rootcert.crt
```

Extract a certificate from a server.

```
# echo | openssl s_client -connect server:443 2>/dev/null | \
  sed -ne '/BEGIN CERT/,/END CERT/p' > svrcert.pem
```

Display the information contained in an X.509 certificate.

```
# openssl x509 -text -noout -in svrcert.pem
```

Display the SHA1 fingerprint of a certificate.

```
# openssl x509 -sha1 -noout -fingerprint -in cert.pem
```

Generate a CSR, writing the unencrypted private key to `prikey.pem` and the request to `csr.pem` for submission to a CA. The CA signs and returns a certificate or a certificate chain that authenticates your public key.

```
# openssl req -new -nodes '/CN=www.unserdom.com/O=Unser Dom, Corp./C=US/ST=Ca/L=Sunnydale' \
  -newkey rsa:1024 -keyout prikey.pem -out csr.pem
```

Display the information contained in a CSR.

```
# openssl req -in csr.pem -noout -text
```

Verify a certificate including the signing authority, signing chain, and period of validity.

```
# openssl verify cert.pem
```

Display the directory that holds information about the CAs trusted by your system. By default, this directory is `/etc/pki/tls`. The `/etc/pki/tls/certs` subdirectory contains trusted certificates.

```
# openssl version -d
```

Create an SHA1 digest of a file.

```
# openssl dgst -sha1 file
```

Sign the SHA1 digest of a file using the private key stored in the file `prikey.pem`.

```
# openssl dgst -sha1 -sign prikey.pem -out file.sha1 file
```

Verify the signed digest for a file using the public key stored in the file `pubkey.pem`.

```
# openssl dgst -sha1 -verify pubkey.pem -signature file.sha1 file
```

List all available ciphers.

```
# openssl list-cipher-commands
```

Encrypt a file using Blowfish.

```
# openssl enc -blowfish -salt -in file -out file.enc
```

Decrypt a Blowfish-encrypted file.

```
# openssl enc -d -blowfish -in file.enc -out file.dec
```

Convert a base 64 encoded certificate (also referred to as PEM or RFC 1421) to binary DER format.

```
# openssl x509 -in cert.pem -outform der -out certificate.der
```

Convert the base 64 encoded certificates for an entity and its CA to a single PKCS7 format certificate.

```
# openssl crl2pkcs7 -nocrl -certfile entCert.cer -certfile CACert.cer -out certificate.p7b
```

For more information, see the `openssl(1)`, `ciphers(1)`, `dgst(1)`, `enc(1)`, `req(1)`, `s_client(1)`, `s_server(1)`, `verify(1)`, and `x509(1)` manual pages.

3.3.2 About the keytool Command

Most Java applications use the *keystore* that is supplied with the JDK to store cryptographic keys, X.509 certificate chain information, and trusted certificates. The default JDK keystore on Oracle Linux is the file `/etc/pki/java/cacerts`. You can use the `keytool` command to generate self-signed certificates and to install and manage certificates in the keystore. Note that the `keytool` command syntax changed in Java SE 6. The examples given here are for that version of `keytool`.

The following are some sample `keytool` commands.

List the contents of the keystore `/etc/pki/java/cacerts`. The default keystore password is `changeit`. If specified, the verbose option `-v` displays detailed information.

```
# keytool -list [-v] -keystore /etc/pki/java/cacerts
```

Change the password for a keystore (for example, `/etc/pki/java/cacerts`).

```
# keytool -storepasswd -keystore /etc/pki/java/cacerts
```

Create a new keystore `keystore.jks` for managing your public/private key pairs and certificates from entities that you trust, generate a public/private key pair using the RSA algorithm and a key length of 1024 bits, and create a self-signed certificate that includes the public key and the specified distinguished name information. *`pkpassword`* is the private key password and *`storepassword`* is the keystore password. The certificate is valid for 100 days and is associated with the private key in a keystore entry that has the alias `engineering`.

```
# keytool -genkeypair -alias mycert -keyalg RSA -keysize 1024 \
-dname "CN=www.unserdom.com, OU=Eng, O=Unser Dom Corp, C=US, ST=Ca, L=Sunnydale" \
-alias engineering -keypass pkpassword -keystore keystore.jks \
-storepass storepassword -validity 100
```

Print the contents of a certificate file in a human-readable form. If specified, the verbose option `-v` displays detailed information.

```
# keytool -printcert [-v] -file cert.cer
```

Generate a CSR in the file `carequest.csr` for submission to a CA. The CA signs and returns a certificate or a certificate chain that authenticates your public key.

```
# keytool -certreq -file carequest.csr
```

Import the root certificate or certificate chain for the CA from the file `ACME.cer` into the keystore `keystore.jks` and give it the alias `acmeca`. If specified, the `-trustcacerts` option instructs `keytool` to add the certificate only if it can validate the chain of trust against the existing root CA certificates in the `cacerts` keystore. Alternatively, use the `keytool -printcert` command to check that the certificate's fingerprint matches the fingerprint that the CA publishes.

```
# keytool -importcert -alias acmeca [-trustcacerts] -file ACME.cer \
  -keystore keystore.jks -storepass storepassword
```

Import the signed certificate for your organization after you have received it from the CA. In this example, the file containing the certificate is `ACMEdom.cer`. The `-alias` option specifies the entry for the first entity in the CA's root certificate chain. The signed certificate is added to the front of the chain and becomes the entity that is addressed by the alias name.

```
# keytool -importcert -v -trustcacerts -alias acmeca -file ACMEdom.cer \
  -keystore keystore.jks -storepass storepassword
```

Delete the certificate with the alias *`aliasname`* from the keystore `keystore.jks`.

```
# keytool -delete -alias aliasname -keystore keystore.jks -storepass storepassword
```

Export the certificate with the alias *`aliasname`* as a binary PKCS7 format file, which includes the supporting certificate chain as well as the issued certificate.

```
# keytool -exportcert -noprompt -alias aliasname -file output.p7b \
  -keystore keystore.jks -storepass storepassword
```

Export the certificate with the alias *`aliasname`* as a base 64 encoded text file (also referred to as PEM or RFC 1421). For a certificate chain. the file includes only the first certificate in the chain, which authenticates the public key of the aliased entity.

```
# keytool -exportcert -noprompt -rfc -alias aliasname -file output.pem \
  -keystore keystore.jks -storepass storepassword
```

For more information, see the `keytool(1)` manual page.

3.4 Configuring and Using Authentication

Authentication is the verification of the identity of a user. A user logs in by providing a user name and a password, and the operating system authenticates the user's identity by comparing this information to data stored on the system. If the login credentials match and the user account is active, the user is authenticated and can successfully access the system.

The information that verifies a user's identity can either be located on the local system in the `/etc/passwd` and `/etc/shadow` files, or on remote systems using Identity Policy Audit (IPA), the Lightweight Directory Access Protocol (LDAP), the Network Information Service (NIS), or Winbind. In addition, IPSv2, LDAP, and NIS data files can use the Kerberos authentication protocol, which allows nodes communicating over a non-secure network to prove their identity to one another in a secure manner.

You can use the Authentication Configuration GUI (`system-config-authentication`) to select the authentication mechanism and to configure any associated authentication options. Alternatively, you can use the `authconfig` command. Both the Authentication Configuration GUI and `authconfig` adjust settings in the PAM configuration files that are located in the `/etc/pam.d` directory.

3.4.1 About Local Oracle Linux Authentication

You can use the User Manager GUI (`system-config-users`) to add or delete users and groups and to modify settings such as passwords, home directories, login shells, and group membership. Alternatively, you can use commands such as `useradd` and `groupadd`.

Unless you select a different authentication mechanism during installation or by using the Authentication Configuration GUI or the `authconfig` command, Oracle Linux verifies a user's identity by using the information that is stored in the `/etc/passwd` and `/etc/shadow` files.

The `/etc/passwd` file stores account information for each user such as his or her unique user ID (or *UID*, which is an integer), user name, home directory, and login shell. A user logs in using his or her user name, but the operating system uses the associated UID. When the user logs in, he or she is placed in his or her home directory and his or her login shell runs.

The `/etc/group` file stores information about groups of users. A user also belongs to one or more groups, and each group can contain one or more users. If you can grant access privileges to a group, all members of the group receive the same access privileges. Each group account has a unique group ID (*GID*, again an integer) and an associated group name.

Oracle Linux implements the *user private group* (*UPG*) scheme where adding a user account also creates a corresponding UPG with the same name as the user, and of which the user is the only member.

Only the `root` user can add, modify, or delete user and group accounts. By default, both users and groups use shadow passwords, which are cryptographically hashed and stored in `/etc/shadow` and `/etc/gshadow` respectively. These shadow password files are readable only by the `root` user. root can set a group password that a user must enter to become a member of the group by using the `newgrp` command. If a group does not have a password, a user can only join the group by `root` adding him or her as a member.

The `/etc/login.defs` file defines parameters for password aging and related security policies.

For more information about the content of these files, see the `group(5)`, `gshadow(5)`, `login.defs(5)`, `passwd(5)`, and `shadow(5)` manual pages.

3.4.2 About IPA

IPA allows you to set up a domain controller for DNS, Kerberos, and authorization policies as an alternative to Active Directory Services. You can enrol client machines with an IPA domain so that they can access information for single sign-on authentication. IPA combines the capabilities of existing well-known technologies such as certificate services, DNS, LDAP, Kerberos, LDAP, and NTP.

To be able to configure IPA authentication, use `yum` to install the `ipa-client` and `ipa-admintools` packages.

If you use the Authentication Configuration GUI and select IPA v2 as the user account database, you are prompted to enter the names of the IPA domain, realm, and server. You can also select to configure NTP so that the system time is consistent with the IPA server. If you have initialized Kerberos, you can click **Join Domain** to create a machine account on the IPA server and grant permission to join the domain.

For more information about configuring IPA, see http://freeipa.org/page/Documentation.

3.4.3 About LDAP Authentication

LDAP allows systems to access centrally stored information over a network. LDAP servers store the information in directory-based database that is optimized for searching. Directory entries are arranged in a hierarchical tree-like structure that can store a variety of information such as names, addresses, phone numbers, authentication data, network services, printers, and many other types of data. LDAP can also be used to authenticate users, allowing users to access their account from any machine on the LDAP network.

An entry is the basic unit of information within an LDAP directory. Each entry has one or more attributes. Each attribute has a name, a type or description, and one or more values. Examples of types are `cn` for common name and `mail` for an email address. In addition, the `objectClass` attribute allows you to control which attributes are required and which are optional. The values of `objectClass` determine the schema rules that an entry must obey.

Each entry in an LDAP directory is uniquely identified and referenced by its Distinguished Name (DN). The DN is constructed by taking the name of the entry itself (called the Relative Distinguished Name or RDN) and concatenating the names of its ancestor entries, known as the LDAP Search Base DN. For example, the DN for a user with an RDN of `uid=gab451` might be similar to `uid=gab451,ou=People,dc=mydomain,dc=com`, where `ou=People,dc=mydomain,dc=com` is the LDAP Search base DN, `ou` stands for Organizational Unit and `dc` stands for Domain Component.

To be able to configure LDAP authentication, use `yum` to install the `openldap-clients` package.

If you use the Authentication Configuration GUI and select LDAP as the user account database, you are prompted to enter the LDAP Search Base DN and the URL of the LDAP server including the port number (for example, `ldap://ldap-svr.mydomain.com:389`).

You can configure LDAP to use either LDAP authentication or Kerberos authentication. LDAP authentication requires that you use either LDAP over SSL (ldaps) or Transport Layer Security (TLS) to secure the connection to the LDAP server. If you use TLS, you must enter the URL from which to download the CA certificate that provides the basis for authentication within the domain.

You can also enable and configure LDAP by using the `authconfig` command.

To use LDAP as the authentication source, specify the `--enableldapauth` option together with the full LDAP server URL (including the port number) and the LDAP Search Base DN, as shown in the following example:.

```
# authconfig --enableldap --enableldapauth \
  --ldapserver=ldap://ldap-svr.mydomain.com:389 \
  --ldapbasedn="ou=people,dc=mydomain,dc=com" \
  --update
```

If you want to use TLS, additionally specify the `--enableldaptls` option and the download URL of the CA certificate:

```
# authconfig --enableldap --enableldapauth \
  --ldapserver=ldap://ldap-svr.mydomain.com:389 \
  --ldapbasedn="ou=people,dc=mydomain,dc=com" \
  --enableldaptls \
  --ldaploadcacert=https://ca-server.mydomain.com/caCert.crt \
  --update
```

For information about using Kerberos authentication with LDAP, see Section 3.4.6, “About Kerberos Authentication”.

For more information, see the `authconfig(8)` manual page.

For more information about LDAP, see the `ldap(3)` manual page.

3.4.4 About NIS Authentication

NIS stores administrative information such as user names, passwords, and host names on a centralized server. Client systems on the network can access this common data. This configuration allows to move from machine to machine without having to remember different passwords and copy data from one machine to another. Storing administrative information centrally, and providing a means of accessing it from networked systems, also ensures the consistency of that data. NIS also reduces the overhead of maintaining administration files such as `/etc/passwd` on each system.

A network of NIS systems is a *NIS domain*. Each system within the domain has the same NIS domain name, which is different from a DNS domain name. The DNS domain is used throughout the Internet to refer to a group of systems. A NIS domain is used to identify systems that use files on a NIS server. A NIS domain must have exactly one master server but can have multiple slave servers.

To be able to configure NIS authentication, use `yum` to install the `yp-tools` and `ypbind` packages.

If you use the Authentication Configuration GUI and select NIS as the user account database, you are prompted to enter the names of the NIS Domain and the NIS master server.

You can configure NIS to use either NIS authentication or Kerberos authentication.

Warning

NIS authentication is deprecated as it has security issues, including a lack of protection of authentication data.

For information about using Kerberos authentication with NIS, see Section 3.4.6, “About Kerberos Authentication”.

3.4.5 About Winbind Authentication

Winbind is a client-side service that resolves user and group information on a Windows server, and allows Oracle Linux to understand Windows users and groups. To be able to configure Winbind authentication, use `yum` to install the `samba-winbind` package. This package includes the `winbindd` daemon that implements the `winbind` service.

If you use the Authentication Configuration GUI and select Winbind as the user account database, you are prompted for the information that is required to connect to a Microsoft workgroup, Active Directory, or Windows NT domain controller. Enter the name of the Winbind domain and select the security model for the Samba server:

`ads`
: In the Activity Directory Server (ADS) security model, Samba acts as a domain member in an ADS realm, and clients use Kerberos tickets for Active Directory authentication. You must configure Kerberos and join the server to the domain, which creates a machine account for your server on the domain controller.

`domain`
: In the domain security model, the local Samba server has a machine account (a domain security trust account) and Samba authenticates user names and passwords with a domain controller in a domain that implements Windows NT4 security.

  Warning

  If the local machine acts as a Primary or Backup Domain Controller, do not use the domain security model. Use the user security model instead.

`server`
: In the server security model, the local Samba server authenticates user names and passwords with another server, such as a Windows NT server.

Warning

The server security model is deprecated as it has numerous security issues.

`user`
: In the user security model, a client must log in with a valid user name and password. This model supports encrypted passwords. If the server successfully validates the client's user name and password, the client can mount multiple shares without being required to specify a password.

Depending on the security model that you choose, you might also need to specify the following information:

- The name of the ADS realm that the Samba server is to join (ADS security model only).
- The names of the domain controllers. If there are several domain controllers, separate the names with spaces.
- The login template shell to use for the Windows NT user account (ADS and domain security models only).
- Whether to allow user authentication using information that has been cached by the System Security Services Daemon (SSSD) if the domain controllers are offline.

Your selection updates the security directive in the `[global]` section of the `/etc/samba/smb.conf` configuration file.

If you have initialized Kerberos, you can click **Join Domain** to create a machine account on the Active Directory server and grant permission for the Samba domain member server to join the domain.

You can also use the `authconfig` command to configure Winbind authentication. To use the user-level security models, specify the name of the domain or workgroup and the host names of the domain controllers. for example:

```
# authconfig --enablewinbind --enablewinbindauth --smbsecurity user \
  [--enablewinbindoffline] --smbservers="ad1.mydomain.com ad2.mydomain.com" \
  --smbworkgroup=MYDOMAIN --update
```

To allow user authentication using information that has been cached by the System Security Services Daemon (SSSD) if the domain controllers are offline, specify the `--enablewinbindoffline` option.

For the domain security model, additionally specify the template shell, for example:

```
# authconfig --enablewinbind --enablewinbindauth --smbsecurity domain \
  [--enablewinbindoffline] --smbservers="ad1.mydomain.com ad2.mydomain.com" \
  --smbworkgroup=MYDOMAIN --update --winbindtemplateshell=/bin/bash --update
```

For the ADS security model, additionally specify the ADS realm and template shell, for example:

```
# authconfig --enablewinbind --enablewinbindauth --smbsecurity ads \
  [--enablewinbindoffline] --smbservers="ad1.mydomain.com ad2.mydomain.com" \
  --smbworkgroup=MYDOMAIN --update --smbrealm MYDOMAIN.COM \
  --winbindtemplateshell=/bin/bash --update
```

For more information, see the `authconfig(8)` manual page.

3.4.6 About Kerberos Authentication

Both LDAP and NIS authentication optionally support Kerberos authentication. (In the case of IPA, Kerberos is fully integrated.) Kerberos provides a secure connection over standard ports, and it also allows offline logins by using credential caching with SSSD.

To be able to use Kerberos authentication, use `yum` to install the `krb5-libs` and `krb5-workstation` packages.

If you use the Authentication Configuration GUI and select LDAP or NIS as the user account database, select Kerberos password as the authentication method. You are prompted for the following information that is required to connect to the Kerberos realm:

- The name of the Kerberos realm.
- A comma-separated list of Key Distribution Center (KDC) servers that can issue Kerberos tickets.
- A comma-separated list of Kerberos Administration Servers.

You can also select whether Kerberos should use DNS to resolve the host names of Kerberos servers and to search for KDCs within the realm. DNS domains are typically coterminous with Kerberos realms.

You can use the following options with the `authconfig` command to configure Kerberos authentication with LDAP or NIS:

`--enablekrb5`	Use Kerberos authentication. (Specify instead of `--enableldapauth` for LDAP.)
`--enablekrb5kdcdns`	Use DNS to resolve the host names of Kerberos servers.
`--enablekrb5realmdns`	Use DNS to search for KDCs within a Kerberos realm.
`--krb5adminserver=`*`server`*	Specify a Kerberos Administration Server.
`--krb5kdc=`*`server`*	Specify a KDC server.
`--krb5realm=`*`realm`*	Specify the name of the Kerberos realm.

For more information, see the `authconfig(8)` manual page.

3.5 Configuring and Using Pluggable Authentication Modules

The Pluggable Authentication Modules (PAM) feature is an authentication mechanism that allows you to configure how applications use authentication to verify the identity of a user. The PAM configuration files, which are located in the `/etc/pam.d` directory, describe the authentication procedure for an application. The name of each configuration file is the same as, or is similar to, the name of the application for which the module provides authentication. For example, the configuration files for `passwd` and `sudo` are named `passwd` and `sudo`.

Each configuration file contains a list (*stack*) of calls to authentication modules. For example, the following is the content of the `login` configuration file:

```
#%PAM-1.0
auth [user_unknown=ignore success=ok ignore=ignore default=bad] pam_securetty.so
auth       include      system-auth
account    required     pam_nologin.so
account    include      system-auth
password   include      system-auth
# pam_selinux.so close should be the first session rule
session    required     pam_selinux.so close
session    required     pam_loginuid.so
session    optional     pam_console.so
# pam_selinux.so open should only be followed by sessions to be executed in the user context
session    required     pam_selinux.so open
session    required     pam_namespace.so
session    optional     pam_keyinit.so force revoke
session     include       system-auth
-session    optional      pam_ck_connector.so
```

Comments in the file start with a # character. The remaining lines each define an operation type, a control flag, the name of a module such as `pam_rootok.so` or the name of an included configuration file such as `system-auth`, and any arguments to the module. PAM provides authentication modules as 32 and 64-bit shared libraries in `/lib/security` and `/lib64/security` respectively.

For a particular operation type, PAM reads the stack from top to bottom and calls the modules listed in the configuration file. Each module generates a success or failure result when called.

The following operation types are defined for use:

`auth`	The module tests whether a user is authenticated or authorized to use a service or application. For example, the module might request and verify a password. Such modules can also set credentials, such as a group membership or a Kerberos ticket.
`account`	The module tests whether an authenticated user is allowed access to a service or application. For example, the module might check if a user account has expired or if a user is allowed to use a service at a given time.
`password`	The module handles updates to an authentication token.
`session`	The module configures and manages user sessions, performing tasks such as mounting or unmounting a user's home directory.

If the operation type is preceded with a dash (`-`), PAM does not add an create a system log entry if the module is missing.

With the exception of `include`, the control flags tell PAM what to do with the result of running a module. The following control flags are defined for use:

`optional`	The module is required for authentication if it is the only module listed for a service.
`required`	The module must succeed for access to be granted. PAM continues to execute the remaining modules in the stack whether the module succeeds or fails. PAM does not immediately inform the user of the failure.
`requisite`	The module must succeed for access to be granted. If the module succeeds, PAM continues to execute the remaining modules in the stack. However, if the module fails, PAM notifies the user immediately and does not continue to execute the remaining modules in the stack.
`sufficient`	If the module succeeds, PAM does not process any remaining modules of the same operation type. If the module fails, PAM processes the remaining modules of the same operation type to determine overall success or failure.

The control flag field can also define one or more rules that specify the action that PAM should take depending on the value that a module returns. Each rule takes the form *`value=action`*, and the rules are enclosed in square brackets, for example:

```
[user_unknown=ignore success=ok ignore=ignore default=bad]
```

If the result returned by a module matches a value, PAM uses the corresponding action, or, if there is no match, it uses the default action.

The `include` flag specifies that PAM must also consult the PAM configuration file specified as the argument.

Most authentication modules and PAM configuration files have their own manual pages. In addition, the `/usr/share/doc/pam-version` directory contains the PAM System Administrator's Guide (`html/Linux-PAM_SAG.html` or `Linux-PAM_SAG.txt`) and a copy of the PAM standard (`rfc86.0.txt`).

For more information, see the `pam(8)` manual page. In addition, each PAM module has its own manual page, for example `pam_unix(8)`.

3.6 Configuring and Using Access Control Lists

POSIX Access Control Lists (ACLs) provide a richer access control model than traditional UNIX Discretionary Access Control (DAC) that sets read, write, and execute permissions for the owner, group, and all other system users. You can configure ACLs that define access rights for more than just a single user or group, and specify rights for programs, processes, files, and directories. If you set a default ACL on a directory, its descendents inherit the same rights automatically. The kernel provides ACL support for `ext3`, `ext4`, and NFS-exported file systems.

The following are examples of setting and displaying ACLs for directories and files.

Grant read access to a file or directory by a user.

```
# setfacl -m u:user:r file
```

Display the name, owner, group, and ACL for a file or directory.

```
# getfacl file
```

Remove write access to a file for all groups and users by modifying the effective rights mask rather than the ACL.

```
# setfacl -m m::rx file
```

Remove the entry for a group from the ACL of a file.

```
# setfacl -x g:group file
```

Copy the ACL of file *f1* to file *f2*.

```
# getfacl f1 | setfacl --set-file=- f2
```

Promote the ACL settings of a directory to default ACL settings that can be inherited.

```
# getfacl --access dir | setfacl -d -M- dir
```

For more information on how to manage ACLs, see the `setfacl(1)` and `getfacl(1)` manual pages.

3.7 Configuring and Using SELinux

Traditional Linux security is based on a Discretionary Access Control (DAC) policy, which provides minimal protection from broken software or from malware that is running as a normal user or as `root`. Access to files and devices is based solely on user identity and ownership. Malware or broken software can do anything with files and resources that the user that started the process can do. If the user is `root` or the application is `setuid` or `setgid` to `root`, the process can have `root`-access control over the entire file system.

The National Security Agency created Security Enhanced Linux (SELinux) to provide a finer-grained level of control over files, processes, users and applications in the Linux operating system. The SELinux enhancement to the Linux kernel implements the Mandatory Access Control (MAC) policy, which allows you to define a security policy that provides granular permissions for all users, programs, processes, files, and devices. The kernel's access control decisions are based on all the security relevant information available, and not solely on the authenticated user identity.

When security-relevant access occurs, such as when a process attempts to open a file, SELinux intercepts the operation in the kernel. If a MAC policy rule allows the operation, it continues; otherwise, SELinux blocks the operation and returns an error to the process. The kernel checks and enforces DAC policy rules before MAC rules, so it does not check SELinux policy rules if DAC rules have already denied access to a resource.

The following table describes the SELinux packages that are installed by default with Oracle Linux:

Package	Description
`policycoreutils`	Provides utilities such as `load_policy`, `restorecon`, `secon`, `setfiles`, `semodule`, `sestatus`, and `setsebool` for operating and managing SELinux.
`libselinux`	Provides the API that SELinux applications use to get and set process and file security contexts, and to obtain security policy decisions.
`selinux-policy`	Provides the SELinux Reference Policy, which is used as the basis for other policies, such as the SELinux targeted policy.
`selinux-policy-targeted`	Provides support for the SELinux targeted policy, where objects outside the targeted domains run under DAC.
`libselinux-python`	Contains Python bindings for developing SELinux applications.
`libselinux-utils`	Provides the `avcstat`, `getenforce`, `getsebool`, `matchpathcon`, `selinuxconlist`, `selinuxdefcon`, `selinuxenabled`, `setenforce`, and `togglesebool` utilities.

The following table describes a selection of useful SELinux packages that are not installed by default:

Package	Description
`mcstrans`	Translates SELinux levels, such as `s0-s0:c0.c1023`, to an easier-to-read form, such as `SystemLow-SystemHigh`.
`policycoreutils-gui`	Provides a GUI (`system-config-selinux`) that you can use to manage SELinux. For example, you can use the GUI to set the system default enforcing mode and policy type.
`policycoreutils-python`	Provides additional Python utilities for operating SELinux, such as `audit2allow`, `audit2why`, `chcat`, and `semanage`.
`selinux-policy-mls`	Provides support for the strict Multilevel Security (MLS) policy as an alternative to the SELinux targeted policy.
`setroubleshoot`	Provides the GUI that allows you to view `setroubleshoot-server` messages using the `sealert` command.
`setroubleshoot-server`	Translates access-denial messages from SELinux into detailed descriptions that you can view on the command line using the `sealert` command.
`setools-console`	Provides the Tresys Technology SETools distribution of tools and libraries, which you can use to analyze and query policies, monitor and report audit logs, and manage file context.

Use `yum` or another suitable package manager to install the SELinux packages that you require on your system.

For more information about SELinux, refer to the SELinux Project Wiki, the `selinux(8)` manual page, and the manual pages for the SELinux commands.

3.7.1 About SELinux Administration

The following table describes the utilities that you can use to administer SELinux, and the packages that contain each utility.

Utility	Package	Description
`audit2allow`	`policycoreutils-python`	Generates SELinux policy `allow_audit` rules from logs of denied operations.
`audit2why`	`policycoreutils-python`	Generates SELinux policy `don't_audit` rules from logs of denied operations.
`avcstat`	`libselinux-utils`	Displays statistics for the SELinux Access Vector Cache (AVC).
`chcat`	`policycoreutils-python`	Changes or removes the security category for a file or user.
`findcon`	`setools-console`	Searches for file context.
`fixfiles`	`policycoreutils`	Fixes the security context for file systems.
`getenforce`	`libselinux-utils`	Reports the current SELinux mode.
`getsebool`	`libselinux-utils`	Reports SELinux boolean values.
`indexcon`	`setools-console`	Indexes file context.
`load_policy`	`policycoreutils`	Loads a new SELinux policy into the kernel.
`matchpathcon`	`libselinux-utils`	Queries the system policy and displays the default security context that is associated with the file path.
`replcon`	`setools-console`	Replaces file context.
`restorecon`	`policycoreutils`	Resets the security context on one or more files.
`restorecond`	`policycoreutils`	Daemon that watches for file creation and sets the default file context.
`sandbox`	`policycoreutils-python`	Runs a command in an SELinux sandbox.
`sealert`	`setroubleshoot-server`, `setroubleshoot`	Acts as the user interface to the `setroubleshoot` system, which diagnoses and explains SELinux AVC denials and provides recommendations on how to prevent such denials.
`seaudit-report`	`setools-console`	Reports from the SELinux audit log.
`sechecker`	`setools-console`	Checks SELinux policies.
`secon`	`policycoreutils`	Displays the SELinux context from a file, program, or user input.
`sediff`	`setools-console`	Compares SELinux polices.
`seinfo`	`setools-console`	Queries SELinux policies.
`selinuxconlist`	`libselinux-utils`	Displays all SELinux contexts that are reachable by a user.
`selinuxdefcon`	`libselinux-utils`	Displays the default SELinux context for a user.
`selinuxenabled`	`libselinux-utils`	Indicates whether SELinux is enabled.

Utility	Package	Description
semanage	policycoreutils-python	Manages SELinux policies.
semodule	policycoreutils	Manages SELinux policy modules.
semodule_deps	policycoreutils	Displays the dependencies between SELinux policy packages.
semodule_expand	policycoreutils	Expands a SELinux policy module package.
semodule_link	policycoreutils	Links SELinux policy module packages together.
semodule_package	policycoreutils	Creates a SELinux policy module package.
sesearch	setools-console	Queries SELinux policies.
sestatus	policycoreutils	Displays the SELinux mode and the SELinux policy that are in use.
setenforce	libselinux-utils	Modifies the SELinux mode.
setsebool	policycoreutils	Sets SELinux boolean values.
setfiles	policycoreutils	Sets the security context for one or more files.
system-config-selinux	policycoreutils-gui	Provides a GUI that you can use to manage SELinux.
togglesebool	libselinux-utils	Flips the current value of an SELinux boolean.

3.7.2 About SELinux Modes

SELinux runs in one of three modes.

Disabled	The kernel uses only DAC rules for access control. SELinux does not enforce any security policy because no policy is loaded into the kernel.
Enforcing	The kernel denies access to users and programs unless permitted by SELinux security policy rules. All denial messages are logged as AVC (Access Vector Cache) denials. This is the default mode that enforces SELinux security policy.
Permissive	The kernel does not enforce security policy rules but SELinux sends denial messages to a log file. This allows you to see what actions would have been denied if SELinux were running in enforcing mode. This mode is intended to used for diagnosing the behavior of SELinux.

3.7.3 Setting SELinux Modes

You can set the default and current SELinux mode in the Status view of the SELinux Administration GUI.

Alternatively, to display the current mode, use the `getenforce` command:

```
# getenforce
Enforcing
```

To set the current mode to `Enforcing`, enter:

```
# setenforce Enforcing
```

To set the current mode to `Permissive`, enter:

```
# setenforce Permissive
```

The current value that you set for a mode using `setenforce` does not persist across reboots. To configure the default SELinux mode, edit the configuration file for SELinux, `/etc/selinux/config`, and set the value of the `SELINUX` directive to `disabled`, `enabled`, or `permissive`.

3.7.4 About SELinux Policies

An SELinux policy describes the access permissions for all users, programs, processes, and files, and for the devices upon which they act. You can configure SELinux to implement either Targeted Policy or Multilevel Security (MLS) Policy.

3.7.4.1 Targeted Policy

Applies access controls to a limited number of processes that are believed to be most likely to be the targets of an attack on the system. Targeted processes run in their own SELinux domain, known as a *confined domain*, which restricts access to files that an attacker could exploit. If SELinux detects that a targeted process is trying to access resources outside the confined domain, it denies access to those resources and logs the denial. Only specific services run in confined domains. Examples are services that listen on a network for client requests, such as `httpd`, `named`, and `sshd`, and processes that run as `root` to perform tasks on behalf of users, such as `passwd`. Other processes, including most user processes, run in an unconfined domain where only DAC rules apply. If an attack compromises an unconfined process, SELinux does not prevent access to system resources and data.

The following table lists examples of SELinux domains.

Domain	Description
`initrc_t`	`init` and processes executed by `init`
`kernel_t`	Kernel processes
`unconfined_t`	Processes executed by Oracle Linux users run in the unconfined domain

3.7.4.2 Multilevel Security (MLS) Policy

Applies access controls to multiple levels of processes with each level having different rules for user access. Users cannot obtain access to information if they do not have the correct authorization to run a process at a specific level. In SELinux, MLS implements the Bell–LaPadula (BLP) model for system security, which applies labels to files, processes and other system objects to control the flow of information between security levels. In a typical implementation, the labels for security levels might range from the most secure, `top secret`, through `secret`, and `classified`, to the least secure, `unclassified`. For example, under MLS, you might configure a program labelled `secret` to be able to write to a file that is labelled `top secret`, but not to be able to read from it. Similarly, you would permit the same program to read from and write to a file labelled `secret`, but only to read `classified` or `unclassified` files. As a result, information that passes through the program can flow upwards through the hierarchy of security levels, but not downwards.

Note

You must install the `selinux-policy-mls` package if you want to be able to apply the MLS policy.

3.7.4.3 Setting SELinux Policies

Note

You cannot change the policy type of a running system.

You can set the default policy type in the **Status** view of the SELinux Administration GUI.

Alternatively, to configure the default policy type, edit `/etc/selinux/config` and set the value of the `SELINUXTYPE` directive to `targeted` or `mls`.

3.7.4.4 Customizing SELinux Policies

You can customize an SELinux policy by enabling or disabling the members of a set of boolean values. Any changes that you make take effect immediately and do not require a reboot.

You can set the boolean values in the **Boolean** view of the SELinux Administration GUI.

Alternatively, to display all boolean values together with a short description, use the following command:

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

ftp_home_dir       (off  ,  off)  Allow ftp to read and write files in the user home ...
smartmon_3ware     (off  ,  off)  Enable additional permissions needed to support dev...
xdm_sysadm_login   (off  ,  off)  Allow xdm logins as sysadm
.
.
.
```

You can use the `getsebool` and `setsebool` commands to display and set the value of a specific boolean.

```
# getsebool boolean
# setsebool boolean on|off
```

For example, to display and set the value of the `ftp_home_dir` boolean:

```
# getsebool ftp_home_dir
ftp_home_dir --> off
# setsebool ftp_home_dir on
# getsebool ftp_home_dir
ftp_home_dir --> on
```

To toggle the value of a boolean, use the `togglesebool` command as shown in this example:

```
# togglesebool ftp_home_dir
ftp_home_dir: inactive
```

To make the value of a boolean persist across reboots, specify the `-P` option to `setsebool`, for example:

```
# setsebool -P ftp_home_dir on
# getsebool ftp_home_dir
ftp_home_dir --> on
```

3.7.5 About SELinux Context

Under SELinux, all file systems, files, directories, devices, and processes have an associated security context. For files, SELinux stores a context label in the extended attributes of the file system. The context contains additional information about a system object: the SELinux user, their role, their type, and the security level. SELinux uses this context information to control access by processes, Linux users, and files.

You can specify the `-Z` option to certain commands (`ls`, `ps`, and `id`) to display the SELinux context with the following syntax:

```
SELinux user:Role:Type:Level
```

where the fields are as follows:

`SELinux user`	An SELinux user account compliments a regular Linux user account. SELinux maps every Linux user to an SELinux user identity that is used in the SELinux context for the processes in a user session.
`Role`	In the Role-Based Access Control (RBAC) security model, a role acts as an intermediary abstraction layer between SELinux process domains or file types and an SELinux user. Processes run in specific SELinux domains, and file system objects are assigned SELinux file types. SELinux users are authorized to perform specified roles, and roles are authorized for specified SELinux domains and file types. A user's role determines which process domains and file types he or she can access, and hence, which processes and files, he or she can access.
`Type`	A type defines an SELinux file type or an SELinux process domain. Processes are separated from each other by running in their own domains. This separation prevents processes from accessing files that other processes use, and prevents processes from accessing other processes. The SELinux policy rules define the access that process domains have to file types and to other process domains.
`Level`	A level is an attribute of Multilevel Security (MLS) and Multicategory Security (MCS). An MLS range is a pair of sensitivity levels, written as *`low_level-high_level`*. The range can be abbreviated as *`low_level`* if the levels are identical. For example, `s0` is the same as `s0-s0`. Each level has an optional set of security categories to which it applies. If the set is contiguous, it can be abbreviated. For example, `s0:c0.c3` is the same as `s0:c0,c1,c2,c3`.

3.7.5.1 Displaying SELinux User Mapping

To display the mapping between SELinux and Linux user accounts, select the User Mapping view in the the SELinux Administration GUI.

Alternatively, enter the following command to display the user mapping:

```
# semanage login -l

Login Name                SELinux User              MLS/MCS Range

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

By default, SELinux maps Linux users other than `root` and the default system-level user, `system_u`, to the Linux `__default__` user, and in turn to the SELinux `unconfined_u` user. The MLS/MCS Range is the security level used by Multilevel Security (MLS) and Multicategory Security (MCS).

3.7.5.2 Displaying SELinux Context Information

To display the context information that is associated with files, use the `ls -Z` command:

```
# ls -Z
-rw-------. root root system_u:object_r:admin_home_t:s0 anaconda-ks.cfg
drwx------. root root unconfined_u:object_r:admin_home_t:s0 Desktop
-rw-r--r--. root root system_u:object_r:admin_home_t:s0 install.log
-rw-r--r--. root root system_u:object_r:admin_home_t:s0 install.log.syslog
```

To display the context information that is associated with a specified file or directory:

```
# ls -Z /etc/selinux/config
-rw-r--r--. root root system_u:object_r:selinux_config_t:s0 /etc/selinux/config
```

To display the context information that is associated with processes, use the `ps -Z` command:

```
# ps -Z
LABEL                                                   PID  TTY   TIME     CMD
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 3038 pts/0 00:00:00 su
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 3044 pts/0 00:00:00 bash
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 3322 pts/0 00:00:00 ps
```

To display the context information that is associated with the current user, use the `id -Z` command:

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

3.7.5.3 Changing the Default File Type

Under some circumstances, you might need to change the default file type for a file system hierarchy. For example, you might want to use a `DocumentRoot` directory other than `/var/www/html` with `httpd`.

To change the default file type of the directory hierarchy `/var/webcontent` as `httpd_sys_content_t`:

1. Use the `semanage` command to define the file type `httpd_sys_content_t` for the directory hierarchy:

   ```
   # /usr/sbin/semanage fcontext -a -t httpd_sys_content_t "/var/webcontent(/.*)?"
   ```

   This command adds the following entry to the file `/etc/selinux/targeted/contexts/files/file_contexts.local`:

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

2. Use the `restorecon` command to apply the new file type to the entire directory hierarchy.

   ```
   # /sbin/restorecon -R -v /var/webcontent
   ```

3.7.5.4 Restoring the Default File Type

To restore the default file type of the directory hierarchy `/var/webcontent` after previously changing it to `httpd_sys_content_t`:

1. Use the `semanage` command to delete the file type definition for the directory hierarchy from the file `/etc/selinux/targeted/contexts/files/file_contexts.local`:

   ```
   # /usr/sbin/semanage fcontext -d "/var/webcontent(/.*)?"
   ```

2. Use the `restorecon` command to apply the default file type to the entire directory hierarchy.

   ```
   # /sbin/restorecon -R -v /var/webcontent
   ```

3.7.5.5 Relabelling a File System

If you see an error message that contains the string `file_t`, the problem usually lies with a file system having an incorrect context label.

To relabel a file system, use one of the following methods:

- In the **Status** view of the SELinux Administration GUI, select the **Relabel on next reboot** option.
- Create the file `/.autorelabel` and reboot the system.
- Run the `fixfiles onboot` command and reboot the system.

3.7.6 About SELinux Users

As described in Section 3.7.5, “About SELinux Context”, each SELinux user account compliments a regular Oracle Linux user account. SELinux maps every Oracle Linux user to an SELinux user identity that is used in the SELinux context for the processes in a user session.

SELinux users form part of a SELinux policy that is authorized for a specific set of roles and for a specific MLS (Multi-Level Security) range, and each Oracle Linux user is mapped to an SELinux user as part of the policy. As a result, Linux users inherit the restrictions and security rules and mechanisms placed on SELinux users. To define the roles and levels of users, the mapped SELinux user identity is used in the SELinux context for processes in a session. You can display user mapping in the **User Mapping** view of the SELinux Administration GUI. You can also view the mapping between SELinux and Oracle Linux user accounts from the command line:

```
# semanage login –l
Login Name    SELinux User     MLS/MCS Range
_default_     unconfined_u     s0-s0:c0.c1023
root          unconfined_u     s0-s0:c0.c1023
system_u      system_u         s0-s0:c0.c1023
```

The MLS/MCS Range column displays the level used by MLS and MCS.

By default, Oracle Linux users are mapped to the SELinux user `unconfined_u`.

You can configure SELinux to confine Oracle Linux users by mapping them to SELinux users in confined domains, which have predefined security rules and mechanisms as listed in the following table.

SELinux User	SELinux Domain	Permit Running su?	Permit Network Access?	Permit Logging in Using X Window System?	Permit Executing Applications in `$HOME` and `/tmp`?
`guest_u`	`guest_t`	No	No	No	No
`staff_u`	`staff_t`	Yes	Yes	Yes	Yes
`user_u`	`user_t`	No	Yes	Yes	Yes
`xguest_x`	`xguest_t`	No	Firefox only	Yes	No

3.7.6.1 Mapping Oracle Linux Users to SELinux Users

To map an Oracle Linux user *`oluser`* to an SELinux user such as `user_u`, use the `semanage` command:

```
# semanage login -a -s user_u oluser
```

3.7.6.2 Configuring the Behavior of Application Execution for Users

To help prevent flawed or malicious applications from modifying a user's files, you can use booleans to specify whether users are permitted to run applications in directories to which they have write access, such as in their home directory hierarchy and `/tmp`.

To allow Oracle Linux users in the `guest_t` and `xguest_t` domains to execute applications in directories to which they have write access:

```
# setsebool -P allow_guest_exec_content on
# setsebool -P allow_xguest_exec_content on
```

To prevent Linux users in the `staff_t` and `user_t` domains from executing applications in directories to which they have write access:

```
# setsebool -P allow_staff_exec_content off
# setsebool -P allow_user_exec_content off
```

3.8 Configuring and Using Auditing

Auditing collects data at the kernel level that you can analyze to identify unauthorized activity. Auditing collects more data in greater detail than system logging, but most audited events are uninteresting and insignificant. The process of examining audit trails to locate events of interest can be a significant challenge that you will probably need to automate.

The audit configuration file, `/etc/audit/auditd.conf`, defines the data retention policy, the maximum size of the audit volume, the action to take if the capacity of the audit volume is exceeded, and the locations of local and remote audit trail volumes. The default audit trail volume is `/var/log/audit/audit.log`. For more information, see the `auditd.conf(5)` manual page.

By default, auditing captures specific events such as system logins, modifications to accounts, and `sudo` actions. You can also configure auditing to capture detailed system call activity or modifications to certain files. The kernel audit daemon (`auditd`) records the events that you configure, including the event type, a time stamp, the associated user ID, and success or failure of the system call.

The entries in the audit rules file, `/etc/audit/audit.rules`, determine which events are audited. Each rule is a command-line option that is passed to the `auditctl` command. You should typically configure this file to match your site's security policy.

The following are examples of rules that you might set in the `/etc/audit/audit.rules` file.

Record all unsuccessful exits from `open` and `truncate` system calls for files in the `/etc` directory hierarchy.

```
-a exit,always -S open -S truncate -F /etc -F success=0
```

Record all files opened by a user with UID 10.

```
-a exit,always -S open -F uid=10
```

Record all files that have been written to or that have their attributes changed by any user who originally logged in with a UID of 500 or greater.

```
-a exit,always -S open -F auid>=500 -F perm=wa
```

Record requests for write or file attribute change access to `/etc/sudoers`, and tag such record with the string `sudoers-change`.

```
-w /etc/sudoers -p wa -k sudoers-change
```

Record requests for write and file attribute change access to the `/etc` directory hierarchy.

```
-w /etc/ -p wa
```

Require a reboot after changing the audit configuration. If specified, this rule should appear at the end of the `/etc/audit/audit.rules` file.

```
-e 2
```

You can find more examples of audit rules in `/usr/share/doc/audit-`*`version`*`/stig.rules`, and in the `auditctl(8)` and `audit.rules(7)` manual pages.

Stringent auditing requirements can impose a significant performance overhead and generate large amounts of audit data. Some site security policies stipulate that a system must shut down if events cannot be recorded because the audit volumes have exceeded their capacity. As a general rule, you should direct audit data to separate file systems in rotation to prevent overspill and to facilitate backups.

You can use the `-k` option to tag audit records so that you can locate them more easily in an audit volume with the `ausearch` command. For example, to examine records tagged with the string `sudoers-change`, you would enter:

```
# ausearch -k sudoers-change
```

The `aureport` command generates summaries of audit data. You can set up `cron` jobs that run `aureport` periodically to generate reports of interest. For example, the following command generates a reports that shows every login event from 1 second after midnight on the previous day until the current time:

```
# aureport -l -i -ts yesterday -te now
```

For more information, see the `ausearch(8)` and `aureport(8)` manual pages.

3.9 Configuring and Using System Logging

The log files contain messages about the system, kernel, services, and applications. For those files that are controlled by the system logging daemon `rsyslogd`, the main configuration file is `/etc/rsyslog.conf`, which contains global directives, module directives, and rules.

Global directives specify configuration options that apply to the `rsyslogd` daemon. All configuration directives must start with a dollar sign (`$`) and only one directive can be specified on each line. The following example specifies the maximum size of the `rsyslog` message queue:

```
$MainMsgQueueSize 50000
```

The available configuration directives are described in the file `/usr/share/doc/rsyslog-`*`version-number`*`/rsyslog_conf_global.html`.

The design of `rsyslog` allows its functionality to be dynamically loaded from modules, which provide configuration directives. To load a module, specify the following directive:

```
$ModLoad MODULE_name
```

Modules have the following main categories:

- Input modules gather messages from various sources. Input module names always start with the `im` prefix (examples include `imfile` and `imrelp`).
- Filter modules allow `rsyslogd` to filter messages according to specified rules. The name of a filter module always starts with the `fm` prefix.
- Library modules provide functionality for other loadable modules. `rsyslogd` loads library modules automatically when required. You cannot configure the loading of library modules.
- Output modules provide the facility to store messages in a database or on other servers in a network, or to encrypt them. Output module names always starts with the `om` prefix (examples include `omsnmp` and `omrelp`).
- Message modification modules change the content of an `rsyslog` message.
- Parser modules allow `rsyslogd` to parse the message content of messages that it receives. The name of a parser module always starts with the `pm` prefix.

- String generator modules generate strings based on the content of messages in cooperation with `rsyslog`'s template feature. The name of a string generator module always starts with the `sm` prefix.

Input modules receive messages, which pass them to one or more parser modules. A parser module creates a representation of a message in memory, possibly modifying the message, and passes the internal representation to output modules, which can also modify the content before outputting the message.

A description of the available modules can be found at http://www.rsyslog.com/doc/rsyslog_conf_modules.html.

An `rsyslog` rule consists of a filter part, which selects a subset of messages, and an action part, which specifies what to do with the selected messages. To define a rule in the `/etc/rsyslog.conf` configuration file, specify a filter and an action on a single line, separated by one or more tabs or spaces.

You can configure rsyslog to filter messages according to various properties. The most commonly used filters are:

- Expression-based filters, written in the `rsyslog` scripting language, select messages according to arithmetic, boolean, or string values.
- Facility/priority-based filters filter messages based on facility and priority values that take the form *`facility.priority`*.
- Property-based filters filter messages by properties such as `timegenerated` or `syslogtag`.

The following table lists the available facility keywords for facility/priority-based filters:

Facility Keyword	Description
`auth`, `authpriv`	Security, authentication, or authorization messages.
`cron`	`crond` messages.
`daemon`	Messages from system daemons other than `crond` and `rsyslogd`.
`kern`	Kernel messages.
`lpr`	Line printer subsystem.
`mail`	Mail system.
`news`	Network news subsystem.
`syslog`	Messages generated internally by `rsyslogd`.
`user`	User-level messages.
`UUCP`	UUCP subsystem.
`local0` - `local7`	Local use.

The following table lists the available priority keywords for facility/priority-based filters, in ascending order of importance:

Priority Keyword	Description
`debug`	Debug-level messages.
`info`	Informational messages.
`notice`	Normal but significant condition.
`warning`	Warning conditions.

Priority Keyword	Description
`err`	Error conditions.
`crit`	Critical conditions.
`alert`	Immediate action required.
`emerg`	System is unstable.

All messages of the specified priority and higher are logged according to the specified action. An asterisk (`*`) wildcard specifies all facilities or priorities. Separate the names of multiple facilities and priorities on a line with commas (`,`). Separate multiple filters on one line with semicolons (;). Precede a priority with an exclamation mark (`!`) to select all messages except those with that priority.

The following are examples of facility/priority-based filters.

Select all kernel messages with any priority.

```
kern.*
```

Select all mail messages with `crit` or higher priority.

```
mail.crit
```

Select all `daemon` and `kern` messages with `warning` or `err` priority.

```
daemon,kern.warning,err
```

Select all `cron` messages except those with `info` or `debug` priority.

```
cron.!info,!debug
```

By default, `/etc/rsyslog.conf` includes the following rules:

```
# Log all kernel messages to the console.
# Logging much else clutters up the screen.
#kern.*                                                 /dev/console

# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;authpriv.none;cron.none                /var/log/messages

# The authpriv file has restricted access.
authpriv.*                                              /var/log/secure

# Log all the mail messages in one place.
mail.*                                                  -/var/log/maillog

# Log cron stuff
cron.*                                                  /var/log/cron

# Everybody gets emergency messages
*.emerg                                                 *

# Save news errors of level crit and higher in a special file.
uucp,news.crit                                          /var/log/spooler

# Save boot messages also to boot.log
local7.*                                                /var/log/boot.log
```

You can send the logs to a central log server over TCP by adding the following entry to the `forwarding rules` section of `/etc/rsyslog.conf` on each log client:

```
*.*        @@logsvr:port
```

where *`logsvr`* is the domain name or IP address of the log server and port is the port number (usually, 514).

On the log server, add the following entry to the `MODULES` section of `/etc/rsyslog.conf`:

```
$ModLoad imtcp
$InputTCPServerRun port
```

where *`port`* corresponds to the port number that you set on the log clients.

To manage the rotation and archival of the correct logs, edit `/etc/logrotate.d/syslog` so that it references each of the log files that are defined in the `RULES` section of `/etc/rsyslog.conf`. You can configure how often the logs are rotated and how many past copies of the logs are archived by editing `/etc/logrotate.conf`.

It is recommended that you configure Logwatch on your log server to monitor the logs for suspicious messages, and disable Logwatch on log clients. However, if you do use Logwatch, disable high precision timestamps by adding the following entry to the GLOBAL DIRECTIVES section of `/etc/rsyslog.conf` on each system:

```
$ActionFileDefaultTemplate RSYSLOG_TraditionalFileFormat
```

For more information, see the `logrotate(8)`, `logwatch(8)`, `rsyslogd(8)` and `rsyslog.conf(5)` manual pages, the HTML documentation in the `/usr/share/doc/rsyslog-5.8.10` directory, and the documentation at http://www.rsyslog.com/doc/manual.html.

3.10 Configuring and Using Process Accounting

The `psacct` package implements the process accounting service in addition to the following utilities that you can use to monitor process activities:

`ac`	Displays connection times in hours for a user as recorded in the `wtmp` file (by default, `/var/log/wtmp`).
`accton`	Turns on process accounting to the specified file. If you do not specify a file name argument, process accounting is stopped. The default system accounting file is `/var/account/pacct`.
`lastcomm`	Displays information about previously executed commands as recorded in the system accounting file.
`sa`	Summarizes information about previously executed commands as recorded in the system accounting file.

Note

As for any logging activity, ensure that the file system has enough space to store the system accounting and `wtmp` files. Monitor the size of the files and, if necessary, truncate them.

For more information, see the `ac(1)`, `accton(8)`, `lastcomm(1)`, and `sa(8)` manual pages.

3.11 Configuring and Using Software Management

Oracle Linux provides the `yum` utility which you can use to install or upgrade RPM packages. The main benefit of using yum is that it also installs or upgrades any package dependencies. `yum` downloads packages from repositories such as those that are available on the Oracle public yum server and the

Unbreakable Linux Network (ULN), but you can also set up your own repositories on systems that do not have Internet access.

The Oracle public yum server is a convenient way to install Oracle Linux packages rather than installing them from installation media. You can also subscribe to the Oracle Linux errata mailing list, and obtain bug fixes, security fixes and enhancements. You can access the server at http://public-yum.oracle.com/.

If you have registered your system with ULN, you can use `yum` with the ULN channels to maintain the software on your system

You can use the RPM package manager to verify the integrity of installed system files. The `rpm -V` ***`package`*** and `rpm -Vf` ***`filename`*** commands verify packages and files respectively by comparing them with package metadata in the RPM database. The verify operation compares file size, MD5 sum, permissions, type, owner, and group and displays any discrepancies. To see more verbose information, specify the `-v` option. You can use the `rpm -qa` command to verify the integrity of all the packages that are installed on a system, for example:

```
# for i in `rpm -qa`
> do
> rpm -V $i > .tmp || echo -e "\nDiscepancies for package $i" && cat .tmp
> rm -f .tmp
> done

Discepancies for package gdm-2.30.4-33.0.1.el6_2.x86_64
.M....G..    /var/log/gdm
.M.......    /var/run/gdm
missing     /var/run/gdm/greeter

Discepancies for package libgcj-4.4.6-4.el6.x86_64
..5....T.  c /usr/lib64/security/classpath.security

Discepancies for package sudo-1.7.4p5-12.el6_3.x86_64
S.5....T.  c /etc/sudoers

Discepancies for package libcgroup-0.37-4.el6.x86_64
S.5....T.  c /etc/cgconfig.conf

Discepancies for package yum-3.2.29-30.0.1.el6.noarch
.......T.  c /etc/yum.conf

Discepancies for package kernel-2.6.32-279.el6.x86_64
.......T.    /etc/ld.so.conf.d/kernel-2.6.32-279.el6.x86_64.conf
...
```

A string of character codes indicates the discrepancies between an installed file and the metadata for that file. The following table lists the meanings of the character codes in the output from `rpm -V`:

Code	Description of Difference
`5`	MD5 sum.
`D`	Device major or minor number.
`G`	Group ownership.
`L`	Symbolic link path.
`M`	Mode including permissions or file type.
`P`	Capabilities.
`S`	File size.
`T`	Modification time.
`U`	User ownership.

Code	Description of Difference
`.`	None (test passed).
`?`	Unknown (test could not be performed).

If displayed, a single character code preceding the affected file denotes the file type, and can take the values shown in the following table:

Code	Description
`c`	Configuration file.
`d`	Documentation file.
`g`	Ghost file, whose file contents are not included in the package payload.
`l`	License file.
`r`	Readme file.

Most discrepancies are caused by editing the configuration files of subsystems. To see which files change over time, create a baseline file of discrepancies immediately after installation, and `diff` this file against the results found by `rpm -V` at a later date.

You can also use a file integrity checker to test whether a system has been compromised. There are several available open source and commercial file integrity checking tools, including AIDE (Advanced Intrusion Detection Environment) and Tripwire. AIDE and Tripwire are intrusion detection systems that scan file systems and record cryptographic hashes of each file in a database. After creating the database, you should then move it to a read-only medium to avoid tampering. On subsequent file system checks, the tool alerts you if the stored checksums do not match those for the current files. For more information, see the AIDE or Tripwire websites.

For more information about using `yum`, see the `yum(8)` manual page and the *Oracle Linux Administrator's Solutions Guide*.

3.11.1 Configuring Update and Patch Management

Effective security practice relies on keeping system software up to date. It is therefore essential to apply system security updates as soon as they are published. It is strongly recommended that you register every IT system with an update management infrastructure. For Oracle Linux systems, the Unbreakable Linux Network (ULN) tracks system software release levels, and advises you as soon as critical updates become available. Updates and errata are also available at no charge from the Oracle public yum repository.

Updating the kernel or core system libraries typically requires a system reboot. In mission-critical enterprise and cloud environments, crucial updates might not get installed until you reboot the systems during a scheduled maintenance window. As a result, systems that support critical business applications could be running while they are not protected from known vulnerabilities. To tackle this problem, Oracle Linux Premier Support includes access to Ksplice Uptrack, which is an innovative technology that allows administrators to apply security updates, patches, and critical bug fixes to the running kernel without requiring a reboot. Ksplice Uptrack improves the security, reliability, and availability of Oracle Linux systems by enabling zero downtime updates, helping to keep systems up to date without downtime or service disruption.

For more information about Ksplice, see https://oss.oracle.com/ksplice/docs/ksplice-quickstart.pdf.

3.11.2 Installing and Using the Yum Security Plugin

The `yum-plugin-security` package allows you to use `yum` to obtain a list of all of the errata that are available for your system, including security updates. You can also use Oracle Enterprise Manager 12c

Cloud Control or management tools such as Katello, Pulp, Red Hat Satellite, Spacewalk, and SUSE Manager to extract and display information about errata.

To install the `yum-plugin-security` package, enter the following command:

```
# yum install yum-plugin-security
```

To list the errata that are available for your system, enter:

```
# yum updateinfo list
Loaded plugins: refresh-packagekit, rhnplugin, security
ELBA-2012-1518 bugfix         NetworkManager-1:0.8.1-34.el6_3.x86_64
ELBA-2012-1518 bugfix         NetworkManager-glib-1:0.8.1-34.el6_3.x86_64
ELBA-2012-1518 bugfix         NetworkManager-gnome-1:0.8.1-34.el6_3.x86_64
ELBA-2012-1457 bugfix         ORBit2-2.14.17-3.2.el6_3.x86_64
ELBA-2012-1457 bugfix         ORBit2-devel-2.14.17-3.2.el6_3.x86_64
ELSA-2013-0215 Important/Sec. abrt-2.0.8-6.0.1.el6_3.2.x86_64
ELSA-2013-0215 Important/Sec. abrt-addon-ccpp-2.0.8-6.0.1.el6_3.2.x86_64
ELSA-2013-0215 Important/Sec. abrt-addon-kerneloops-2.0.8-6.0.1.el6_3.2.x86_64
ELSA-2013-0215 Important/Sec. abrt-addon-python-2.0.8-6.0.1.el6_3.2.x86_64
ELSA-2013-0215 Important/Sec. abrt-cli-2.0.8-6.0.1.el6_3.2.x86_64
ELSA-2013-0215 Important/Sec. abrt-desktop-2.0.8-6.0.1.el6_3.2.x86_64
...
```

The output from the command sorts the available errata in order of their IDs, and it also specifies whether each erratum is a security patch (*`severity`*`/Sec.`), a bug fix (`bugfix`), or a feature enhancement (`enhancement`). Security patches are listed by their severity: `Important`, `Moderate`, or `Low`.

You can use the `--sec-severity` option to filter the security errata by severity, for example:

```
# yum updateinfo list --sec-severity=Moderate
Loaded plugins: refresh-packagekit, rhnplugin, security
ELSA-2013-0269 Moderate/Sec. axis-1.2.1-7.3.el6_3.noarch
ELSA-2013-0668 Moderate/Sec. boost-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-date-time-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-devel-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-filesystem-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-graph-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-iostreams-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-program-options-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-python-1.41.0-15.el6_4.x86_64
...
```

To list the security errata by their Common Vulnerabilities and Exposures (CVE) IDs instead of their errata IDs, specify the keyword `cves` as an argument:

```
# yum updateinfo list cves
Loaded plugins: refresh-packagekit, rhnplugin, security
 CVE-2012-5659 Important/Sec. abrt-2.0.8-6.0.1.el6_3.2.x86_64
 CVE-2012-5660 Important/Sec. abrt-2.0.8-6.0.1.el6_3.2.x86_64
 CVE-2012-5659 Important/Sec. abrt-addon-ccpp-2.0.8-6.0.1.el6_3.2.x86_64
 CVE-2012-5660 Important/Sec. abrt-addon-ccpp-2.0.8-6.0.1.el6_3.2.x86_64
 CVE-2012-5659 Important/Sec. abrt-addon-kerneloops-2.0.8-6.0.1.el6_3.2.x86_64
 CVE-2012-5660 Important/Sec. abrt-addon-kerneloops-2.0.8-6.0.1.el6_3.2.x86_64
 CVE-2012-5659 Important/Sec. abrt-addon-python-2.0.8-6.0.1.el6_3.2.x86_64
 CVE-2012-5660 Important/Sec. abrt-addon-python-2.0.8-6.0.1.el6_3.2.x86_64
...
```

Similarly, the keywords `bugfix`, `enhancement`, and `security` filter the list for all bug fixes, enhancements, and security errata.

You can use the `--cve` option to display the errata that correspond to a specified CVE, for example:

```
# yum updateinfo list --cve CVE-2012-2677
Loaded plugins: refresh-packagekit, rhnplugin, security
```

```
ELSA-2013-0668 Moderate/Sec. boost-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-date-time-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-devel-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-filesystem-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-graph-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-iostreams-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-program-options-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-python-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-regex-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-serialization-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-signals-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-system-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-test-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-thread-1.41.0-15.el6_4.x86_64
ELSA-2013-0668 Moderate/Sec. boost-wave-1.41.0-15.el6_4.x86_64
updateinfo list done
```

To display more information, specify `info` instead of `list`, for example:

```
# yum updateinfo info --cve CVE-2012-2677
Loaded plugins: refresh-packagekit, rhnplugin, security
===============================================================================
   boost security update
===============================================================================
  Update ID : ELSA-2013-0668
    Release : Oracle Linux 6
       Type : security
     Status : final
     Issued : 2013-03-21
       CVEs : CVE-2012-2677
Description : [1.41.0-15]
            : - Add in explicit dependences between some boost
            :   subpackages
            :
            : [1.41.0-14]
            : - Build with -fno-strict-aliasing
            :
            : [1.41.0-13]
            : - In Boost.Pool, be careful not to overflow
            :   allocated chunk size (boost-1.41.0-pool.patch)
            :
            : [1.41.0-12]
            : - Add an upstream patch that fixes computation of
            :   CRC in zlib streams.
            : - Resolves: #707624
   Severity : Moderate
updateinfo info done
```

To update all packages for which security-related errata are available to the latest versions of the packages, even if those packages include bug fixes or new features but not security errata, enter:

```
# yum --security update
```

To update all packages to the latest versions that contain security errata, ignoring any newer packages that do not contain security errata, enter:

```
# yum --security update-minimal
```

To update all kernel packages to the latest versions that contain security errata, enter:

```
# yum --security update-minimal kernel*
```

You can also update only those packages that correspond to a CVE or erratum, for example:

```
# yum update --cve CVE-2012-3954
# yum update --advisory ELSA-2012-1141
```

Note

Some updates might require you to reboot the system. By default, the boot manager will automatically enable the most recent kernel version.

For more information, see the `yum-security(8)` manual page.

3.12 Configuring Access to Network Services

As networks are usually the primary point of entry point into IT systems, you can use network intrusion prevention and detection tools to help avert or uncover a security breach. You can then take steps such as disabling unused network services and configure a packet-filtering firewall and TCP wrappers.

There are several open-source tools for performing packet logging and analysis. For example, tcpdump and Snort capture TCP traffic and analyze it for suspicious usage patterns, such as those that typically occur with port scans or network DoS attacks. Sguil incorporates tcpdump, Snort, and the Wireshark protocol analyzer to provide a network intrusion and detection system that simplifies log analysis and reporting.

You can check what services are running on a system by using port scanning utilities. The following examples show the information that the `netstat`, `lsof`, and `nmap` commands return about open TCP ports and the associated services:

```
# netstat -tulp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address              Foreign Address   State    PID/Program name
tcp        0      0 localhost:ipp              *:*               LISTEN   1657/cupsd
tcp        0      0 localhost:smtp             *:*               LISTEN   1987/master
tcp        0      0 localhost:29754            *:*               LISTEN   2072/vpnagentd
tcp        0      0 *:amqp                     *:*               LISTEN   2030/qpidd
tcp        0      0 *:56652                    *:*               LISTEN   1605/rpc.statd
tcp        0      0 *:sunrpc                   *:*               LISTEN   1542/rpcbind
tcp        0      0 *:ssh                      *:*               LISTEN   1887/sshd
tcp        0      0 localhost:ipp              *:*               LISTEN   1657/cupsd
tcp        0      0 localhost:smtp             *:*               LISTEN   1987/master
tcp        0      0 *:45534                    *:*               LISTEN   1605/rpc.statd
tcp        0      0 *:amqp                     *:*               LISTEN   2030/qpidd
tcp        0      0 *:sunrpc                   *:*               LISTEN   1542/rpcbind
tcp        0      0 localhost:47314            *:*               LISTEN   2873/java
tcp        0      0 *:ssh                      *:*               LISTEN   1887/sshd
udp        0      0 *:bootpc                   *:*                        1584/dhclient
udp        0      0 *:44127                    *:*                        1605/rpc.statd
udp        0      0 *:sunrpc                   *:*                        1542/rpcbind
udp        0      0 10.0.2.15:ntp              *:*                        1895/ntpd
udp        0      0 localhost:ntp              *:*                        1895/ntpd
udp        0      0 *:ntp                      *:*                        1895/ntpd
udp        0      0 *:mdns                     *:*                        1580/avahi-daemon
udp        0      0 *:ipp                      *:*                        1657/cupsd
udp        0      0 *:869                      *:*                        1542/rpcbind
udp        0      0 *:33669                    *:*                        1580/avahi-daemon
udp        0      0 *:933                      *:*                        1605/rpc.statd
udp        0      0 *:sunrpc                   *:*                        1542/rpcbind
udp        0      0 localhost:ntp              *:*                        1895/ntpd
udp        0      0 fe80::a00:27ff:fe16:c333:ntp *:*                      1895/ntpd
udp        0      0 *:ntp                      *:*                        1895/ntpd
udp        0      0 *:44822                    *:*                        1605/rpc.statd
udp        0      0 *:869                      *:*                        1542/rpcbind

# lsof -iTCP -sTCP:LISTEN
COMMAND    PID    USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
rpcbind   1542     rpc    8u  IPv4  11032      0t0  TCP *:sunrpc (LISTEN)
rpcbind   1542     rpc   11u  IPv6  11037      0t0  TCP *:sunrpc (LISTEN)
rpc.statd 1605 rpcuser    9u  IPv4  11201      0t0  TCP *:56652 (LISTEN)
rpc.statd 1605 rpcuser   11u  IPv6  11207      0t0  TCP *:45534 (LISTEN)
cupsd     1657    root    6u  IPv6  12375      0t0  TCP localhost:ipp (LISTEN)
cupsd     1657    root    7u  IPv4  12376      0t0  TCP localhost:ipp (LISTEN)
sshd      1887    root    3u  IPv4  13541      0t0  TCP *:ssh (LISTEN)
sshd      1887    root    4u  IPv6  13543      0t0  TCP *:ssh (LISTEN)
master    1987    root   12u  IPv4  13081      0t0  TCP localhost:smtp (LISTEN)
master    1987    root   13u  IPv6  13083      0t0  TCP localhost:smtp (LISTEN)
qpidd     2030   qpidd   10u  IPv4  13257      0t0  TCP *:amqp (LISTEN)
qpidd     2030   qpidd   11u  IPv6  13258      0t0  TCP *:amqp (LISTEN)
vpnagentd 2072    root   15u  IPv4  13823      0t0  TCP localhost:29754 (LISTEN)
java      2873   guest    7u  IPv6  20694      0t0  TCP localhost:47314 (LISTEN)

# nmap -sTU 10.0.2.15

Starting Nmap 5.51 ( http://nmap.org ) at 2012-12-10 09:37 GMT
Nmap scan report for 10.0.2.15
Host is up (0.0017s latency).
Not shown: 1993 closed ports
PORT     STATE         SERVICE
22/tcp   open          ssh
111/tcp  open          rpcbind
68/udp   open|filtered dhcpc
111/udp  open          rpcbind
123/udp  open          ntp
631/udp  open|filtered ipp
5353/udp open|filtered zeroconf

Nmap done: 1 IP address (1 host up) scanned in 12.66 seconds
```

For more information, see the `lsof(8)`, `netstat(8)`, and `nmap(1)` manual pages.

Caution

Before installing or using the `nmap` command, check the local legislation relating to port scanning software. In some jurisdictions, the possession or use of port scanning software is considered as unlawful criminal activity. Some ISPs might also have acceptable use policies that forbid using such software outside of your private networks.

3.12.1 Configuring and Using Packet-filtering Firewalls

A packet filtering firewall filters incoming and outgoing network packets based on the packet header information. You can create packet filter rules that determine whether packets are accepted or rejected. For example, if you create a rule to block a port, any request is made to that port that is blocked by the firewall, and the request is ignored. Any service that is listening on a blocked port is effectively disabled.

The Oracle Linux kernel uses the Netfilter feature to provide packet filtering functionality for IPv4 and IPv6 packets respectively.

Netfilter consist of two components:

- A `netfilter` kernel component consisting of a set of tables in memory for the rules that the kernel uses to control network packet filtering.
- The `iptables` and `ip6tables` utilities to create, maintain, and display the rules that `netfilter` stores.

To implement a simple, general-purpose firewall, you can use the Firewall Configuration GUI (`system-config-firewall`) to create basic Netfilter rules. To create a more complex firewall configuration, use the `iptables` and `ip6tables` utilities to configure the packet filtering rules.

Netfilter records the packet filtering rules in the `/etc/sysconfig/iptables` and `/etc/sysconfig/ip6tables` files, which `netfilter` reads when it is initialized.

The `netfilter` tables include:

`Filter`	The default table, which is mainly used to drop or accept packets based on their content.
`Mangle`	This table is used to alter certain fields in a packet.
`NAT`	The Network Address Translation table is used to route packets that create new connections.

The kernel uses the rules stored in these tables to make decisions about network packet filtering. Each rule consists of one or more criteria and a single action. If a criterion in a rule matches the information in a network packet header, the kernel applies the action to the packet. Examples of actions include:

`ACCEPT`	Continue processing the packet.
`DROP`	End the packet's life without notice.
`REJECT`	As `DROP`, and additionally notify the sending system that the packet was blocked.

Rules are stored in chains, where each chain is composed of a default policy plus zero or more rules. The kernel applies each rule in a chain to a packet until a match is found. If there is no matching rule, the kernel applies the chain's default action (policy) to the packet.

Each `netfilter` table has several predefined chains. The `filter` table contains the following chains:

`FORWARD`	Packets that are not addressed to the local system pass through this chain.
`INPUT`	Inbound packets to the local system pass through this chain.
`OUTPUT`	Locally created packets pass through this chain.

The chains are permanent and you cannot delete them. However, you can create additional chains in the filter table.

For more information, see the `iptables(8)` and `ip6tables(8)` manual pages.

3.12.1.1 Listing Firewall Rules

Use the `iptables -L` command to list firewall rules for the chains of the `filter` table. The following example shows the default rules for a newly installed system:

```
# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source       destination
ACCEPT     all  --  anywhere     anywhere        state RELATED,ESTABLISHED
ACCEPT     icmp --  anywhere     anywhere
ACCEPT     all  --  anywhere     anywhere
ACCEPT     tcp  --  anywhere     anywhere        state NEW tcp dpt:ssh
ACCEPT     udp  --  anywhere     anywhere        state NEW udp dpt:ipp
ACCEPT     udp  --  anywhere     224.0.0.251     state NEW udp dpt:mdns
ACCEPT     tcp  --  anywhere     anywhere        state NEW tcp dpt:ipp
ACCEPT     udp  --  anywhere     anywhere        state NEW udp dpt:ipp
REJECT     all  --  anywhere     anywhere        reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
target     prot opt source       destination
REJECT     all  --  anywhere     anywhere        reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)
target     prot opt source       destination
```

In this example, the default policy for each chain is `ACCEPT`. A more secure system could have a default policy of `DROP`, and the additional rules would only allow specific packets on a case-by-case basis.

If you want to modify the chains, specify the `--line-numbers` option to see how the rules are numbered.

```
# iptables -L --line-numbers
Chain INPUT (policy ACCEPT)
num  target     prot opt source           destination
1    ACCEPT     all  --  anywhere         anywhere        state RELATED,ESTABLISHED
2    ACCEPT     icmp --  anywhere         anywhere
3    ACCEPT     all  --  anywhere         anywhere
4    ACCEPT     tcp  --  anywhere         anywhere        state NEW tcp dpt:ssh
5    ACCEPT     udp  --  anywhere         anywhere        state NEW udp dpt:ipp
6    ACCEPT     udp  --  anywhere         224.0.0.251     state NEW udp dpt:mdns
7    ACCEPT     tcp  --  anywhere         anywhere        state NEW tcp dpt:ipp
8    ACCEPT     udp  --  anywhere         anywhere        state NEW udp dpt:ipp
9    REJECT     all  --  anywhere         anywhere        reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
num  target     prot opt source           destination
1    REJECT     all  --  anywhere         anywhere        reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)
num  target     prot opt source           destination
```

3.12.1.2 Inserting Rules in a Chain

Use the `iptables -I` command to insert a rule in a chain. For example, the following command inserts a rule in the `INPUT` chain to allow access by TCP on port 80:

```
# iptables -I INPUT 4 -p tcp -m tcp --dport 80 -j ACCEPT
# iptables -L --line-numbers
Chain INPUT (policy ACCEPT)
num  target     prot opt source           destination
1    ACCEPT     all  --  anywhere         anywhere        state RELATED,ESTABLISHED
2    ACCEPT     icmp --  anywhere         anywhere
3    ACCEPT     all  --  anywhere         anywhere
4    ACCEPT     tcp  --  anywhere         anywhere        tcp dpt:http
5    ACCEPT     tcp  --  anywhere         anywhere        state NEW tcp dpt:ssh
6    ACCEPT     udp  --  anywhere         anywhere        state NEW udp dpt:ipp
7    ACCEPT     udp  --  anywhere         224.0.0.251     state NEW udp dpt:mdns
8    ACCEPT     tcp  --  anywhere         anywhere        state NEW tcp dpt:ipp
9    ACCEPT     udp  --  anywhere         anywhere        state NEW udp dpt:ipp
10   REJECT     all  --  anywhere         anywhere        reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
num  target     prot opt source           destination
1    REJECT     all  --  anywhere         anywhere        reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)
num  target     prot opt source           destination
```

The output from `iptables -L` shows that the new entry has been inserted as rule 4, and the old rules 4 through 9 are pushed down to positions 5 through 10. The TCP destination port of 80 is represented as `http`, which corresponds to the following definition in the `/etc/services` file (the HTTP daemon listens for client requests on port 80):

```
http            80/tcp          www www-http    # WorldWideWeb HTTP
```

3.12.1.3 Deleting Rules in a Chain

Use the `iptables -D` command to delete a rule in a chain. For example, the following command deletes rule 4 from the `INPUT` chain:

```
# iptables -D INPUT 4
```

3.12.2 Configuring and Using TCP Wrappers

TCP wrappers provide basic filtering of incoming network traffic. You can allow or deny access from other systems to certain *wrapped* network services running on a Linux server. A wrapped network service is one that has been compiled against the `libwrap.a` library. You can use the `ldd` command to determine if a network service has been wrapped as shown in the following example for the `sshd` daemon:

```
# ldd /usr/sbin/sshd | grep libwrap
 libwrap.so.0 => /lib64/libwrap.so.0 (0x00007f877de07000)
```

When a remote client attempts to connect to a network service on the system, the wrapper consults the rules in the configuration files `/etc/hosts.allow` and `/etc/hosts.deny` files to determine if access is permitted.

The wrapper for a service first reads `/etc/hosts.allow` from top to bottom. If the daemon and client combination matches an entry in the file, access is allowed. If the wrapper does not find a match in `/etc/hosts.allow`, it reads `/etc/hosts.deny` from top to bottom. If the daemon and client combination matches and entry in the file, access is denied. If no rules for the daemon and client combination are found in either file, or if neither file exists, access to the service is allowed.

The wrapper first applies the rules specified in `/etc/hosts.allow`, so these rules take precedence over the rules specified in `/etc/hosts.deny`. If a rule defined in `/etc/hosts.allow` permits access to a service, any rule in `/etc/hosts.deny` that forbids access to the same service is ignored.

The rules take the following form:

```
daemon_list : client_list [: command] [: deny]
```

where *`daemon_list`* and *`client_list`* are comma-separated lists of daemons and clients, and the optional *`command`* is run when a client tries to access a daemon. You can use the keyword `ALL` to represent all daemons or all clients. Subnets can be represented by using the `*` wildcard, for example `192.168.2.*`. Domains can be represented by prefixing the domain name with a period (`.`), for example `.mydomain.com`. The optional `deny` keyword causes a connection to be denied even for rules specified in the `/etc/hosts.allow` file.

The following are some sample rules.

Match all clients for `scp`, `sftp`, and `ssh` access (`sshd`).

```
sshd : ALL
```

Match all clients on the 192.168.2 subnet for FTP access (`vsftpd`).

```
vsftpd : 192.168.2.*
```

Match all clients in the `mydomain.com` domain for access to all wrapped services.

```
ALL : .mydomain.com
```

Match all clients for FTP access, and displays the contents of the banner file `/etc/banners/vsftpd` (the banner file must have the same name as the daemon).

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

Match all clients on the 200.182.68 subnet for all wrapped services, and logs all such events. The `%c` and `%d` tokens are expanded to the names of the client and the daemon.

```
ALL : 200.182.68.* : spawn /bin/echo `date` “Attempt by %c to connect to %d" >> /var/log/tcpwr.log
```

Match all clients for `scp`, `sftp`, and `ssh` access, and logs the event as an `emerg` message, which is displayed on the console.

```
sshd : ALL : severity emerg
```

Match all clients in the `forbid.com` domain for `scp`, `sftp`, and `ssh` access, logs the event, and deny access (even if the rule appears in `/etc/hosts.allow`).

```
sshd : .forbid.com : spawn /bin/echo `date` "sshd access denied for %c" >>/var/log/sshd.log : deny
```

For more information, see the `hosts_access(5)` manual page.

3.13 Configuring and Using Chroot Jails

A `chroot` operation changes the apparent root directory for a running process and its children. It allows you to run a program with a root directory other than `/`. The program cannot see or access files outside the designated directory tree. Such an artificial root directory is called a *chroot jail*, and its purpose is to limit the directory access of a potential attacker. The chroot jail locks down a given process and any user ID that it is using so that all they see is the directory in which the process is running. To the process, it appears that the directory in which it is running is the root directory.

Note

The `chroot` mechanism cannot defend against intentional tampering or low-level access to system devices by privileged users. For example, a `chroot root` user could create device nodes and mount file systems on them. A program can also break out of a chroot jail if it can gain `root` privilege and use `chroot()` to change its current working directory to the real `root` directory. For this reason, you should ensure that a chroot jail does not contain any `setuid` or `setgid` executables that are owned by `root`.

For a `chroot` process to be able to start successfully, you must populate the `chroot` directory with all required program files, configuration files, device nodes, and shared libraries at their expected locations relative to the level of the `chroot` directory.

3.13.1 Running DNS and FTP Services in a Chroot Jail

If the DNS name service daemon (`named`) runs in a chroot jail, any hacker that enters your system via a BIND exploit is isolated to the files under the chroot jail directory. Installing the `bind-chroot` package creates the `/var/named/chroot` directory, which becomes the chroot jail for all BIND files.

You can configure the `vsftpd` FTP server to automatically start chroot jails for clients. By default, anonymous users are placed in a chroot jail. However, local users that access an `vsftpd` FTP server are placed in their home directory. Specify the `chroot_local_user=YES` option in the `/etc/vsftpd/vsftpd.conf` file to place local users in a chroot jail based on their home directory.

3.13.2 Creating a Chroot Jail

To create a chroot jail:

1. Create the directory that will become the root directory of the chroot jail, for example:

   ```
   # mkdir /home/oracle/jail
   ```

2. Use the `ldd` command to find out which libraries are required by the command that you intend to run in the chroot jail, for example `/bin/bash`:

   ```
   # ldd /bin/bash
    linux-vdso.so.1 =>  (0x00007fff56fcc000)
    libtinfo.so.5 => /lib64/libtinfo.so.5 (0x0000003ad1200000)
    libdl.so.2 => /lib64/libdl.so.2 (0x0000003abe600000)
    libc.so.6 => /lib64/libc.so.6 (0x0000003abe200000)
    /lib64/ld-linux-x86-64.so.2 (0x0000003abde00000)
   ```

3. Create subdirectories of the chroot jail's root directory that have the same relative paths as the command binary and its required libraries have to the real root directory, for example:

   ```
   # mkdir /home/oracle/jail/bin
   # mkdir /home/oracle/jail/lib64
   ```

4. Copy the binary and the shared libraries to the directories under the chroot jail's root directory, for example:

   ```
   # cp /bin/bash /home/oracle/jail/bin
   # cp /lib64/{libtinfo.so.5,libdl.so.2,libc.so.6,ld-linux-x86-64.so.2} \
     /home/oracle/jail/lib64
   ```

3.13.3 Using a Chroot Jail

To run a command in a chroot jail in an existing directory (*`chroot_jail`*), use the following command:

```
# chroot chroot_jail command
```

If you do not specify a command argument, `chroot` runs the value of the `SHELL` environment variable or `/bin/sh` if `SHELL` is not set.

For example, to run `/bin/bash` in a chroot jail (having previously set it up as described in Section 3.13.2, “Creating a Chroot Jail”):

```
# chroot /home/oracle/jail
bash-4.1# pwd
/
bash-4.1# ls
bash: ls: command not found
bash-4.1# exit
exit
#
```

You can run built-in shell commands such as `pwd` in this shell, but not other commands unless you have copied their binaries and any required shared libraries to the chroot jail.

For more information, see the `chroot(1)` manual page.

3.14 Configuring and Using Linux Containers

Note

Linux Containers (LXC) are available in Oracle Linux 6 with the Unbreakable Enterprise Kernel (2.6.29 or above). LXC is a Technology Preview feature that is made available for testing and evaluation purposes, but is not recommended for production systems.

The LXC feature provides a way to isolate a group of processes from other processes that are running on an Oracle Linux system. LXC is a lightweight operating system virtualization technology that uses the control group (cgroup) feature to provide resource management and namespace isolation in a similar manner to `chroot`. Within a container, processes can have their own private view of the operating system with its own process ID space, file system structure, and network interfaces.

See the *Oracle Linux Administrator's Solutions Guide* for more information on how to configure and use Linux Containers.

3.15 Configuring and Using Kernel Security Mechanisms

The Linux kernel features some additional security mechanisms that you can use to enhance the security of a system. These mechanisms randomize the layout of a process's address space or prevent code from being executed in non-executable memory.

3.15.1 Address Space Layout Randomization

Address Space Layout Randomization (ASLR) can help defeat certain types of buffer overflow attacks. ASLR can locate the base, libraries, heap, and stack at random positions in a process's address space, which makes it difficult for an attacking program to predict the memory address of the next instruction. ASLR is built into the Linux kernel and is controlled by the parameter `/proc/sys/kernel/randomize_va_space`. The `randomize_va_space` parameter can take the following values:

0	Disable ASLR. This setting is applied if the kernel is booted with the `norandmaps` boot parameter.
1	Randomize the positions of the stack, virtual dynamic shared object (VDSO) page, and shared memory regions. The base address of the data segment is located immediately after the end of the executable code segment.
2	Randomize the positions of the stack, VDSO page, shared memory regions, and the data segment. This is the default setting.

You can change the setting temporarily by writing a new value to `/proc/sys/kernel/randomize_va_space`, for example:

```
# echo value > /proc/sys/kernel/randomize_va_space
```

To change the value permanently, add the setting to `/etc/sysctl.conf`, for example:

```
kernel.randomize_va_space = value
```

and run the `sysctl -p` command.

If you change the value of `randomize_va_space`, you should test your application stack to ensure that it is compatible with the new setting.

If necessary, you can disable ASLR for a specific program and its child processes by using the following command:

```
% setarch `uname -m` -R program [args ...]
```

3.15.2 Data Execution Prevention

The Data Execution Prevention (DEP) feature prevents an application or service from executing code in a non-executable memory region. Hardware-enforced DEP works in conjunction with the NX (Never eXecute) bit on compatible CPUs. Oracle Linux does not emulate the NX bit in software for CPUs that do not implement the NX bit in hardware.

You cannot disable the DEP feature.

3.15.3 Position Independent Executables

The Position Independent Executables (PIE) feature loads executable binaries at random memory addresses so that the kernel can disallow text relocation. To generate a position-independent binary:

- Specify the `-fpie` option to `gcc` when compiling.
- Specify the `-pie` option to `ld` when linking.

To test whether a binary or library is relocatable, use the following command:

```
# readelf -d elfname | grep TEXTREL
```

Chapter 4 Security Considerations for Developers

Table of Contents

4.1 Design Principles for Secure Coding 51
4.2 General Guidelines for Secure Coding 52
4.3 General Guidelines for Network Programs 53

This chapter provides information for developers about how to create secure applications for Oracle Linux, and how to extend Oracle Linux to access external systems without compromising security.

4.1 Design Principles for Secure Coding

The following well-established design principles are recommended for secure coding:

Least privilege	A process or user should be given only those privileges that are necessary to complete a task. User privileges should be assigned according to their role, but not otherwise. To create a minimal protection domain, assign rights when a process or thread requires them and remove them afterwards. This principle limits the potential damage that can result from attacks and user errors.
Economy of mechanism	Keep the design simple. There is less to go wrong, fewer inconsistencies are possible, and the code is easier to understand and debug.
Complete mediation	Check every attempt to access to a resource, not just the first. For example, Linux checks access permissions when a process opens a file but not thereafter. If a file's permissions change while a process has the file open, unauthorized access can result. Ideally, one could argue that the permissions should be checked whenever an open file is accessed. In practise, such checking is considered to be an unnecessary overhead given the circumstances under which access was first obtained.
Open design	Security should not depend on the secrecy of the code's design or implementation, sometimes referred to as *security through obscurity*. For example, an open back door to a system is only as secure as the knowledge of its existence. Of course, this principle does not apply to information such as passwords or cryptographic keys, knowledge of which should also be shared among as few people as possible. For this reason, many secure authentication schemes also rely on biometric identification or the possession of a physical artifact such a hardware token or smart card, in addition to knowledge of a PIN code or password.
Separation of privilege	Divide the code into modules, where each module requires a specific, limited set of privileges to perform a specific task. Typically, multiple privileges should be required to grant access to a sensitive operation. This principle ensures separation of duty and provides defense in depth. For example, a main thread that has no privileges can generate a privileged thread to perform a task. A successful attack against the main thread thus gains minimal access to the system.

Least common mechanism	A system should isolate users and their activities from each other. Users should not share processes or threads and information channels should not be shared between users.
Fail-safe defaults	The default action should be to deny access to an operation. Should an attempt to perform an operation be denied, the system is as secure as it was before the operation started.
Accountability	Log the user and their privileges for each action that he or she attempts to perform. Any logs should be capable of being rotated and archived to avoid filling up a file system.
Psychological acceptability	Security mechanisms should be easy to install, configure, and use so that a user is less tempted to try to bypass them.

4.2 General Guidelines for Secure Coding

The following coding practices are recommended:

- Check that input data is what the program expects by performing type, length, and bound checking. Inputs include command-line arguments and environment variables in addition to data that a user enters.
- Check input data for the inclusion of constructs such as shell commands, SQL statements, and XML and HTML code that might be used in an injection attack.
- Check the type, length, and bounds of arguments to system calls and library routines. If possible, use library routines that guard against buffer overflows.
- Use full pathnames for file-name arguments, do not use files in world-writable directories, verify that a file being written to is not actually a symbolic link, and protect against the unintended overwriting of existing files.
- Check the type, length, and bounds of values returned by system calls and library routines. Make the code respond appropriately to any error codes that system calls and library functions set or return.
- Do not assume the state of the shell environment. Check any settings that a program inherits from the shell, such as the user file-creation mask, signal handling, file descriptors, current working directory, and environment variables, especially `PATH` and `IFS` . Reset the settings if necessary.
- Perform assert checking on variables that can take a finite set of values.
- Log information about privileged actions and error conditions. Do not allow the program to dump a core file on an end-user system.
- Do not echo passwords to the screen, or transmit or store them as clear text. Before transmitting or storing a password, combine it with a salt value and use a secure one-way algorithm such as SHA-512 to create a hash.
- If your program uses a pseudo-random number generating routine, verify that the numbers that it generates are sufficiently random for your requirements. You should also use a good random seed that a potential attacker should not be able to predict. See RFC 4086, *Randomness Requirements for Security*, for more information.
- It is recommended that Address Space Layout Randomization (ASLR) is enabled on the host system as this feature can help defeat certain types of buffer overflow attacks. See Section 3.15.1, “Address Space Layout Randomization”.

- When compiling and linking your program, use the Position Independent Executables (PIE) feature to generate a position-independent binary. See Section 3.15.3, “Position Independent Executables”.
- Consider using `chroot()` to confine the operating boundary of your program to a specified location within a file system.
- Do not execute a shell command by calling `popen()` or `syscall()` from within a program, especially from a `setuid` or `setgid` program.

The following guidelines apply if your program has its `setuid` or `setgid` bit set so that it can perform privileged actions on behalf of a user who does not possess those privileges:

- Do not set the `setuid` or `setgid` bit on shell scripts. However, if you use Perl scripts that are `setuid` or `setgid`, `perl` runs in *taint mode*, which is claimed to be more secure than using the equivalent C code. See the `perlsec(1)` manual page for details.
- Restrict the use of the privilege that `setuid` or `setgid` grants to the functionality that requires it, and then return the effective UID or GID to that of the user. If possible, perform the privileged functionality in a separate, closely-monitored thread or process.
- Do not allow a `setuid` or `setgid` program to execute child processes using `execlp()` or `execvp()`, which use the `PATH` environment variable.

4.3 General Guidelines for Network Programs

The following coding practices are recommended for network programs:

- Perform a reverse lookup on an IP address to obtain the fully qualified domain name, and then use that domain name look up the IP address. The two IP addresses should be identical.
- Protect a service against Denial of Service (DoS) attacks by allowing it to stop processing requests if it becomes overloaded.
- Set timeouts on read and write requests over the network.
- Check the content, bounds, value, and type of data received over the network, and reject any data that does not conform to what the program expects.
- Use certificates or preshared keys to authenticate the local and remote ends of the network connection.
- Use a well-established technology such as TLS or SSL to encrypt data sent over the network connection.
- Wherever possible, use existing networking protocols and technologies whose security characteristics are well known.
- Log information about successful and unsuccessful connection attempts, data reception and transmission errors, and changes to the service state.

Chapter 5 Secure Deployment Checklist

Table of Contents

5.1 Minimizing the Software Footprint 55
5.2 Configuring System Logging 56
5.3 Disabling Core Dumps 57
5.4 Minimizing Active Services 57
5.5 Locking Down Network Services 60
5.6 Configuring a Packet-filtering Firewall 61
5.7 Configuring TCP Wrappers 61
5.8 Configuring Kernel Parameters 61
5.9 Restricting Access to SSH Connections 62
5.10 Configuring File System Mounts, File Permissions, and File Ownerships 62
5.11 Checking User Accounts and Privileges 64

The sections in this chapter provide guidelines that help secure your Oracle Linux system.

5.1 Minimizing the Software Footprint

On systems on which Oracle Linux has been installed, remove unneeded RPMs to minimize the software footprint. For example, you could uninstall the X Windows package (`xorg-x11-server-Xorg`) if it is not required on a server system.

To discover which package provides a given command or file, use the `yum provides` command as shown in the following example:

```
# yum provides /usr/sbin/sestatus
...
policycoreutils-2.0.83-19.24.0.1.el6.x86_64 : SELinux policy core utilities
Repo        : installed
Matched from:
Other       : Provides-match: /usr/sbin/sestatus
```

To display the files that a package provides, use the `repoquery` utility, which is included in the `yum-utils` package. For example, the following command lists the files that the `btrfs-progs` package provides.

```
# repoquery -l btrfs-progs
/sbin/btrfs
/sbin/btrfs-convert
/sbin/btrfs-debug-tree
.
.
.
```

To uninstall a package, use the `yum remove` command, as shown in this example:

```
# yum remove xinetd
Loaded plugins: refresh-packagekit, security
Setting up Remove Process
Resolving Dependencies
--> Running transaction check
---> Package xinetd.x86_64 2:2.3.14-35.el6_3 will be erased
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package        Arch           Version                    Repository          Size
================================================================================
Removing:
 xinetd         x86_64         2:2.3.14-35.el6_3          @ol6_latest        259 k

Transaction Summary
================================================================================
Remove        1 Package(s)

Installed size: 259 k
Is this ok [y/N]: y
Downloading Packages:
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Erasing    : 2:xinetd-2.3.14-35.el6_3.x86_64                              1/1
  Verifying  : 2:xinetd-2.3.14-35.el6_3.x86_64                              1/1

Removed:
  xinetd.x86_64 2:2.3.14-35.el6_3

Complete!
```

The following table lists packages that you should not install or that you should remove using `yum remove` if they are already installed.

Package	Description
`krb5-appl-clients`	Kerberos versions of `ftp`, `rcp`, `rlogin`, `rsh` and `telnet`. If possible, use SSH instead.
`rsh`, `rsh-server`	`rcp`, `rlogin`, and `rsh` use unencrypted communication that can be snooped. Use SSH instead.
`samba`	Network services used by Samba. Remove this package if the system is not acting as an Active Directory server, a domain controller, or as a domain member, and it does not provide Microsoft Windows file and print sharing functionality.
`talk`, `talk-server`	`talk` is considered obsolete.
`telnet`, `telnet-server`	`telnet` uses unencrypted communication that can be snooped. Use SSH instead.
`tftp`, `tftp-server`	TFTP uses unencrypted communication that can be snooped. Use only if required to support legacy hardware. If possible, use SSH or other secure protocol instead.
`xinetd`	The security model used by the Internet listener daemon is deprecated.
`ypbind`, `ypserv`	The security model used by NIS is inherently flawed. Use an alternative such as LDAP or Kerberos instead.

5.2 Configuring System Logging

Verify that the system logging service `rsyslog` is running:

```
# service rsyslog status
rsyslogd (pid  1632) is running...
```

If the service is not running, start it and enable it to start when the system is rebooted:

```
# service rsyslog start
# chkconfig rsyslog on
```

Ensure that each log file referenced in `/etc/rsyslog.conf` exists and is owned and only readable by `root`:

```
# touch logfile
# chown root:root logfile
# chmod 0600 logfile
```

It is also recommended that you use a central log server and that you configure Logwatch on that server. See Section 3.9, "Configuring and Using System Logging".

5.3 Disabling Core Dumps

Core dumps can contain information that an attacker might be able to exploit and they take up a large amount of disk space. To prevent the system creating core dumps when the operating system terminates a program due to a segment violation or other unexpected error, add the following line to `/etc/security/limits.conf`:

```
*   hard   core   0
```

You can restrict access to core dumps to certain users or groups, as described in the `limits.conf(5)` manual page.

By default, the system prevents `setuid` and `setgid` programs, programs that have changed credentials, and programs whose binaries do not have read permission from dumping core. To ensure that the setting is permanently recorded, add the following lines to `/etc/sysctl.conf`:

```
# Disallow core dumping by setuid and setgid programs
fs.suid_dumpable = 0
```

and then run the `sysctl -p` command.

Note

A value of 1 permits core dumps that are readable by the owner of the dumping process. A value of 2 permits core dumps that are readable only by `root` for debugging purposes.

5.4 Minimizing Active Services

Restrict services to only those that a server requires. The default installation for an Oracle Linux server configures a minimal set of services:

`cupsd` and `lpd` (print services)

`sendmail` (email delivery service)

`sshd` (openSSH services)

If possible, configure one type of service per physical machine, virtual machine, or Linux Container. This technique limits exposure if a system is compromised.

If a service is not used, remove the software packages that are associated with the service. If it is not possible to remove a service because of software dependencies, use the `chkconfig` and `service` commands to disable the service.

For services that are in use, apply the latest Oracle support patches and security updates to keep software packages up to date. To protect against unauthorized changes, ensure that the `/etc/services` file is owned by `root` and writable only by `root`.

```
# ls -Z /etc/services
-rw-r--r--. root root system_u:object_r:etc_t:SystemLow /etc/services
```

Unless specifically stated otherwise, consider disabling the services in the following table if they are not used on your system:

Service	Description
`anacron`	Executes commands periodically. Primarily intended for use on laptop and user desktop machines that do not run continuously.
`apmd`	(Advanced Power Management Daemon) Provides information on power management and battery status, and allows programmed response to power management events. Primarily intended for use on laptop machines.
`automount`	Manages mount points for the automatic file-system mounter. Disable this service on servers that do not require automounter functionality.
`bluetooth`	Supports the connections of Bluetooth devices. Primarily intended for use on laptop and user desktop machines. Bluetooth provides an additional potential attack surface. Disable this service on servers that do not require Bluetooth functionality.
`firstboot`	Configures a system when you first log in after installation. Controlled by the `/etc/rc.d/init.d/firstboot` script. `firstboot` does not run unless `RUN_FIRSTBOOT=YES` is set in `/etc/sysconfig/firstboot`. If `/etc/reconfigSys` exists or if you specify `reconfig` in the kernel boot arguments, `firstboot` runs in reconfiguration mode. Disable this service on servers following successful installation.
`gpm`	(General Purpose Mouse) Provides support for the mouse pointer in a text console.
`haldaemon`	(Hardware Abstraction Layer Daemon) Maintains a real-time database of the devices that are connected to a system. Applications can use the HAL API to discover and interact with newly attached devices. Primarily intended for use on laptop and user desktop machines to support hot-plug devices. ⚠ **Caution** Do not disable this service. Many applications rely on this functionality.
`hidd`	(Bluetooth Human Interface Device daemon) Provides support for Bluetooth input devices such as a keyboard or mouse. Primarily intended for use on laptop and user desktop machines. Bluetooth provides an additional potential attack surface. Disable this service on servers that do not require Bluetooth functionality.
`irqbalance`	Distributes hardware interrupts across processors on a multiprocessor system. Disable this service on servers that do not require this functionality.
`iscsi`	Controls logging in to iSCSI targets and scanning of iSCSI devices. Disable this service on servers that do not access iSCSI devices.
`iscsid`	Implements control and management for the iSCSI protocol. Disable this service on servers that do not access iSCSI devices.
`kdump`	Allows a `kdump` kernel to be loaded into memory at boot time or a kernel dump to be saved if the system panics. Disable this service on servers that you do not use for debugging or testing.
`mcstrans`	Controls the SELinux Context Translation System service.

Service	Description
`mdmonitor`	Checks the status of all software RAID arrays on the system. Disable this service on servers that do not use software RAID.
`messagebus`	Broadcasts notifications of system events and other messages relating to hardware events via the system-wide D-BUS message bus. ⚠ **Caution** Do not disable this service. Many applications rely on this functionality.
`microcode_ctl`	Runs microcode that is required for IA32 processors only. Disable this service on servers that do not have such processors.
`pcscd`	(PC/SC Smart Card Daemon) Supports communication with smart-card readers. Primarily intended for use on laptop and user desktop machines to support smart-card authentication. Disable this service on servers that do not use smart-card authentication.
`sandbox`	Sets up `/tmp`, `/var/tmp`, and home directories to be used with the `pam_namespace`, `sandbox`, and `xguest` application confinement utilities. Disable this service if you do not use these programs.
`setroubleshoot`	Controls the SELinux Troubleshooting service, which provides information about SELinux Access Vector Cache (AVC) denials to the `sealert` tool.
`smartd`	Communicates with the Self-Monitoring, Analysis and Reporting Technology (SMART) systems that are integrated into many ATA-3 and later, and SCSI-3 disk drives. SMART systems monitor disk drives to measure reliability, predict disk degradation and failure, and perform drive testing.
`xfs`	Caches fonts in memory to improve the performance of X Window System applications.

You should consider disabling the following network services if they are not used on your system:

Service	Description
`avahi-daemon`	Implements Apple's Zero configuration networking (also known as Rendezvous or Bonjour). Primarily intended for use on laptop and user desktop machines to support music and file sharing. Disable this service on servers that do not require this functionality.
`cups`	Implements the Common UNIX Printing System. Disable this service on servers that do not need to provide this functionality.
`hplip`	Implements HP Linux Imaging and Printing to support faxing, printing, and scanning operations on HP inkjet and laser printers. Disable this service on servers that do not require this functionality.
`isdn`	(Integrated Services Digital Network) Provides support for network connections over ISDN devices. Disable this service on servers that do not directly control ISDN devices.
`netfs`	Mounts and unmounts network file systems, including NCP, NFS, and SMB. Disable this service on servers that do not require this functionality.
`network`	Activates all network interfaces that are configured to start at boot time.
`NetworkManager`	Switches network connections automatically to use the best connection that is available.

Service	Description
`nfslock`	Implements the Network Status Monitor (NSM) used by NFS. Disable this service on servers that do not require this functionality.
`nmb`	Provides NetBIOS name services used by Samba. Disable this service and remove the `samba` package if the system is not acting as an Active Directory server, a domain controller, or as a domain member, and it does not provide Microsoft Windows file and print sharing functionality.
`portmap`	Implements Remote Procedure Call (RPC) support for NFS. Disable this service on servers that do not require this functionality.
`rhnsd`	Queries the Unbreakable Linux Network (ULN) for updates and information.
`rpcgssd`	Used by NFS. Disable this service on servers that do not require this functionality.
`rpcidmapd`	Used by NFS. Disable this service on servers that do not require this functionality.
`smb`	Provides SMB network services used by Samba. Disable this service and remove the `samba` package if the system is not acting as an Active Directory server, a domain controller, or as a domain member, and it does not provide Microsoft Windows file and print sharing functionality.

To stop a service and prevent it from starting when you reboot the system, used the following commands:

```
# service service_name stop
# chkconfig service_name off
```

Alternatively, use the Service Configuration GUI (`system-config-services`) to configure services.

5.5 Locking Down Network Services

Note

It is recommended that you do not install the `xinetd` Internet listener daemon. If you do not need this service, remove the package altogether by using the `yum remove xinetd` command.

If you must enable `xinetd` on your system, minimize the network services that `xinetd` can launch by disabling those services that are defined in the configuration files in `/etc/xinetd.d` and which are not needed.

To counter potential Denial of Service (DoS) attacks, you can configure the resource limits for such services by editing `/etc/xinetd.conf` and related configuration files. For example, you can set limits for the connection rate, the number of connection instances to a service, and the number of connections from an IP address:

```
# Maximum number of connections per second and
# number of seconds for which a service is disabled
# if the maximum number of connections is exceeded
cps             = 50 10

# Maximum number of connections to a service
instances       = 50

# Maximum number of connections from an IP address
per_source      = 10
```

For more information, see the `xinetd(8)` and `xinetd.conf(5)` manual pages.

5.6 Configuring a Packet-filtering Firewall

You can configure the Netfilter feature to act as a packet-filtering firewall that uses rules to determine whether network packets are received, dropped, or forwarded.

The primary interfaces for configuring the packet-filter rules are the `iptables` and `ip6tables` utilities and the Firewall Configuration Tool GUI (`system-config-firewall`). By default, the rules should drop any packets that are not destined for a service that the server hosts or that originate from networks other than those to which you want to allow access.

In addition, Netfilter provides Network Address Translation (NAT) to hide IP addresses behind a public IP address, and IP masquerading to alter IP header information for routed packets. You can also set rule-based packet logging and define a dedicated log file in `/etc/syslog.conf`.

For more information, see Section 3.12.1, “Configuring and Using Packet-filtering Firewalls”.

5.7 Configuring TCP Wrappers

The TCP wrappers feature mediates requests from clients to services, and control access based on rules that you define in the `/etc/hosts.deny` and `/etc/hosts.allow` files. You can restrict and permit service access for specific hosts or whole networks. A common way of using TCP wrappers is to detect intrusion attempts. For example, if a known malicious host or network attempts to access a service, you can deny access and send a warning message about the event to a log file or to the system console.

For more information, see Section 3.12.2, “Configuring and Using TCP Wrappers”.

5.8 Configuring Kernel Parameters

You can use several kernel parameters to counteract various kinds of attack.

`kernel.randomize_va_space` controls Address Space Layout Randomization (ASLR), which can help defeat certain types of buffer overflow attacks. A value of 0 disables ASLR, 1 randomizes the positions of the stack, virtual dynamic shared object (VDSO) page, and shared memory regions, and 2 randomizes the positions of the stack, VDSO page, shared memory regions, and the data segment. The default and recommended setting is 2.

`net.ipv4.conf.all.accept_source_route` controls the handling of source-routed packets, which might have been generated outside the local network. A value of 0 rejects such packets, and 1 accepts them. The default and recommended setting is 0.

`net.ipv4.conf.all.rp_filter` controls reversed-path filtering of received packets to counter IP address spoofing. A value of 0 disables source validation, 1 causes packets to be dropped if the routing table entry for their source address does not match the network interface on which they arrive, and 2 causes packets to be dropped if source validation by reversed path fails (see RFC 1812). The default setting is 0. A value of 2 can cause otherwise valid packets to be dropped if the local network topology is complex and RIP or static routes are used.

`net.ipv4.icmp_echo_ignore_broadcasts` controls whether ICMP broadcasts are ignored to protect against Smurf DoS attacks. A value of 1 ignores such broadcasts, and 0 accepts them. The default and recommended setting is 1.

`net.ipv4.icmp_ignore_bogus_error_message` controls whether ICMP bogus error message responses are ignored. A value of 1 ignores such messages, and 0 accepts them. The default and recommended setting is 1.

To change the value of a kernel parameter, add the setting to `/etc/sysctl.conf`, for example:

```
kernel.randomize_va_space = 1
```

and then run the `sysctl -p` command.

5.9 Restricting Access to SSH Connections

The Secure Shell (SSH) allows protected, encrypted communication with other systems. As SSH is an entry point into the system, disable it if it is not required, or alternatively, edit the `/etc/ssh/sshd_config` file to restrict its use.

For example, the following setting does not allow `root` to log in using SSH:

```
PermitRootLogin no
```

You can restrict remote access to certain users and groups by specifying the `AllowUsers`, `AllowGroups`, `DenyUsers`, and `DenyGroups` settings, for example:

```
DenyUsers carol dan
AllowUsers alice bob
```

The `ClientAliveInterval` and `ClientAliveCountMax` settings cause the SSH client to time out automatically after a period of inactivity, for example:

```
# Disconnect client after 300 seconds of inactivity
ClientAliveCountMax 0
ClientAliveInterval 300
```

After making changes to the configuration file, restart the `sshd` service for your changes to take effect.

For more information, see the `sshd_config(5)` manual page.

5.10 Configuring File System Mounts, File Permissions, and File Ownerships

Use separate disk partitions for operating system and user data to prevent a *file system full* issue from impacting the operation of a server. For example, you might create separate partitions for `/home`, `/tmp`, `p`, `/oracle`, and so on.

Establish disk quotas to prevent a user from accidentally or intentionally filling up a file system and denying access to other users.

To prevent the operating system files and utilities from being altered during an attack, mount the `/usr` file system read-only. If you need to update any RPMs on the file system, use the `-o remount,rw` option with the `mount` command to remount `/usr` for both read and write access. After performing the update, use the `-o remount,ro` option to return the `/usr` file system to read-only mode.

To limit user access to non-`root` local file systems such as `/tmp` or removable storage partitions, specify the `-o noexec, nosuid, nodev` options to `mount`. These option prevent the execution of binaries (but not scripts), prevent the `setuid` bit from having any effect, and prevent the use of device files.

Use the `find` command to check for unowned files and directories on each file system, for example:

```
# find mount_point -mount -type f -nouser -o -nogroup -exec ls -l {} \;
# find mount_point -mount -type d -nouser -o -nogroup -exec ls -l {} \;
```

Unowned files and directories might be associated with a deleted user account, they might indicate an error with software installation or deleting, or they might a sign of an intrusion on the system. Correct the permissions and ownership of the files and directories that you find, or remove them. If possible, investigate and correct the problem that led to their creation.

Use the `find` command to check for world-writable directories on each file system, for example:

```
# find mount_point -mount -type d -perm /o+w -exec ls -l {} \;
```

Investigate any world-writable directory that is owned by a user other than a system user. The user can remove or change any file that other users write to the directory. Correct the permissions and ownership of the directories that you find, or remove them.

You can also use `find` to check for `setuid` and `setgid` executables.

```
# find path -type f \( -perm -4000 -o -perm -2000 \) -exec ls -l {} \;
```

If the `setuid` and `setgid` bits are set, an executable can perform a task that requires other rights, such as `root` privileges. However, buffer overrun attacks can exploit such executables to run unauthorized code with the rights of the exploited process.

If you want to stop a `setuid` and `setgid` executable from being used by non-`root` users, you can use the following commands to unset the `setuid` or `setgid` bit:

```
# chmod u-s file
# chmod g-s file
```

For example, you could use the `chmod` command to unset the `setuid` bit for the `/bin/ping6` command:

```
# ls -al /bin/ping6
-rwsr-xr-x. 1 root root 36488 May 20  2011 /bin/ping6
# chmod u-s /bin/ping6
# ls -al /bin/ping6
-rwxr-xr-x. 1 root root 36488 May 20  2011 /bin/ping6
```

The following table lists programs for which you might want to consider unsetting `setuid` and `setgid`:

Program File	Bit Set	Description of Usage
`/bin/ping`	`setuid`	Sends an ICMP `ECHO_REQUEST` to a network host.
`/bin/ping6`	`setuid`	Sends an ICMPv6 `ECHO_REQUEST` to a network host.
`/bin/cgexec`	`setgid`	Runs a task in a control group.
`/sbin/mount.nfs`	`setuid`	Mounts an NFS file system. **Note** `/sbin/mount.nfs4`, `/sbin/umount.nfs`, and `/sbin/umount.nfs4` are symbolic links to this file.
`/sbin/netreport`	`setgid`	Requests notification of changes to network interfaces.
`/usr/bin/chage`	`setuid`	Finds out password aging information (via the `-l` option).
`/usr/bin/chfn`	`setuid`	Changes `finger` information.
`/usr/bin/chsh`	`setuid`	Changes the login shell.
`/usr/bin/crontab`	`setuid`	Edits, lists, or removes a `crontab` file.

Program File	Bit Set	Description of Usage
`/usr/bin/wall`	`setgid`	Sends a system-wide message.
`/usr/bin/write`	`setgid`	Sends a message to another user.
`/usr/bin/Xorg`	`setuid`	Invokes the X Windows server.
`/usr/libexec/openssh/ssh-keysign`	`setuid`	Runs the SSH helper program for host-based authentication.
`/usr/sbin/suexec`	`setuid`	Switches user before executing external CGI and SSI programs. This program is intended to be used by the Apache HTTP server. For more information, see http://httpd.apache.org/docs/2.2/suexec.html.
`/usr/sbin/usernetctl`	`setuid`	Controls network interfaces. Permission for a user to alter the state of a network inerface also requires `USERCTL=yes` to be set in the interface file. You can also grant users and groups the privilege to run the `ip` command by creating a suitable entry in the `/etc/sudoers` file.

Note

This list is not exhaustive as many optional packages contain `setuid` and `setgid` programs.

5.11 Checking User Accounts and Privileges

Check the system for unlocked user accounts on a regular basis, for example using a command such as the following:

```
# for u in `cat /etc/passwd | cut -d: -f1 | sort`; do passwd -S $u; done
abrt LK 2012-06-28 0 99999 7 -1 (Password locked.)
adm LK 2011-10-13 0 99999 7 -1 (Alternate authentication scheme in use.)
apache LK 2012-06-28 0 99999 7 -1 (Password locked.)
avahi LK 2012-06-28 0 99999 7 -1 (Password locked.)
avahi-autoipd LK 2012-06-28 0 99999 7 -1 (Password locked.)
bin LK 2011-10-13 0 99999 7 -1 (Alternate authentication scheme in use.)
...
```

In the output from this command, the second field shows if a user account is locked (`LK`), does not have a password (`NP`), or has a valid password (`PS`). The third field shows the date on which the user last changed their password. The remaining fields show the minimum age, maximum age, warning period, and inactivity period for the password and additional information about the password's status. The unit of time is days.

Use the `passwd` command to set passwords on any accounts that are not protected.

Use `passwd -l` to lock unused accounts. Alternatively, use `userdel` to remove the accounts entirely.

For more information, see the `passwd(1)` and `userdel(8)` manual pages.

To specify how users' passwords are aged, edit the following settings in the `/etc/login.defs` file:

Setting	Description
`PASS_MAX_DAYS`	Maximum number of days for which a password can be used before it must be changed. The default value is 99,999 days.
`PASS_MIN_DAYS`	Minimum number of days that is allowed between password changes. The default value is 0 days.

Setting	Description
`PASS_WARN_AGE`	Number of days warning that is given before a password expires. The default value is 7 days.

For more information, see the `login.defs(5)` manual page.

To change how long a user's account can be inactive before it is locked, use the `usermod` command. For example, to set the inactivity period to 30 days:

```
# usermod -f 30 username
```

To change the default inactivity period for new user accounts, use the `useradd` command:

```
# useradd -D -f 30
```

A value of -1 specifies that user accounts are not locked due to inactivity.

For more information, see the `useradd(8)` and `usermod(8)` manual pages.

Verify that no user accounts other than `root` have a user ID of 0.

```
# awk -F":" '$3 == 0 { print $1 }' /etc/passwd
root
```

If you install software that creates a default user account and password, change the vendor's default password immediately. Centralized user authentication using an LDAP implementation such as OpenLDAP can help to simplify user authentication and management tasks, and also reduces the risk arising from unused accounts or accounts without a password.

By default, an Oracle Linux system is configured so that you cannot log in directly as `root`. You must log in as a named user before using either `su` or `sudo` to perform tasks as `root`. This configuration allows system accounting to trace the original login name of any user who performs a privileged administrative action. If you want to grant certain users authority to be able to perform specific administrative tasks via `sudo`, use the `visudo` command to modify the `/etc/sudoers` file. For example, the following entry grants the user `erin` the same privileges as `root` when using `sudo`, but defines a limited set of privileges to `frank` so that he can run commands such as `chkconfig`, `service`, `rpm`, and `yum`:

```
erin            ALL=(ALL)       ALL
frank           ALL= SERVICES, SOFTWARE
```

Oracle Linux supports the pluggable authentication modules (PAM) feature, which makes it easier to enforce strong user authentication and password policies, including rules for password complexity, length, age, expiration and the reuse of previous passwords. You can configure PAM to block user access after too many failed login attempts, after normal working hours, or if too many concurrent sessions are opened.

PAM is highly customizable by its use of different modules with customisable parameters. For example, the default password integrity checking module `pam_cracklib.so` tests password strength. The PAM configuration file (`/etc/pam.d/system-auth`) contains the following default entries for testing a password's strength:

```
password  requisite   pam_cracklib.so try_first_pass retry=3 type=
password  sufficient  pam_unix.so sha512 shadow nullok try_first_pass use_authtok
password  required    pam_deny.so
```

The line for `pam_cracklib.so` defines that a user gets three attempts to choose a good password. From the module's default settings, the password length must a minimum of six characters, of which three characters must be different from the previous password.

The line for `pam_unix.so` specifies that the module is not to perform password checking (`pam_cracklib` will already have performed such checks), to use SHA-512 password hashing, to allow access if the existing password is null, and to use the `/etc/shadow` file.

You can modify the control flags and module parameters to change the checking that is performed when a user changes his or her password, for example:

```
password  required  pam_cracklib.so retry=3 minlen=8 difok=5 minclass=-1
password  required  pam_unix.so use_authtok sha512 shadow remember=5
password  required  pam_deny.so
```

The line for `pam_cracklib.so` defines that a user gets three attempts to choose a good password with a minimum of eight characters, of which five characters must be different from the previous password, and which must contain at least one upper case letter, one lower case letter, one numeric digit, and one non-alphanumeric character.

The line for `pam_unix.so` specifies that the module is not to perform password checking, to use SHA-512 password hashing, to use the `/etc/shadow` file, and to save information about the previous five passwords for each user in the `/etc/security/opasswd` file. As `nullok` is not specified, a user cannot change his or her password if the existing password is null.

The omission of the `try_first_pass` keyword means that the user is always asked for their existing password, even if he or she entered it for the same module or for a previous module in the stack.

Alternative modules are available for password checking, such as `pam_passwdqc.so`.

For more information, see Section 3.5, “Configuring and Using Pluggable Authentication Modules” and the `pam_cracklib(8)`, `pam_deny(8)`, `pam_passwdqc(8)`, and `pam_unix(8)` manual pages.

Chapter 6 Enabling FIPS Mode for OpenSSL

Table of Contents

6.1 About the FIPS 140-2 OpenSSL Library Modules 67
6.2 Configuring a System for FIPS Mode 67
6.3 Using the FIPS-Compliant OpenSSL Library 69

This chapter describes how to make OpenSSL on an Oracle Linux 6 Update 5 or later system compliant with Federal Information Processing Standard (FIPS) Publication 140-2, which is a standard that establishes security requirements for cryptographic modules.

For detailed information about FIPS, see http://csrc.nist.gov/publications/PubsFIPS.html.

6.1 About the FIPS 140-2 OpenSSL Library Modules

The following FIPS 140-2 validated OpenSSL library modules are available in the `openssl-fips-1.0.1*` package for installation on Oracle Linux 6 Update 5 and later for the x86-64 architecture:

- `/usr/lib64/libcrypto.so.1.0.1*` (64-bit library)
- `/usr/lib64/libssl.so.1.0.1*` (64-bit library)

6.2 Configuring a System for FIPS Mode

Note

The system must have been registered with ULN. The `openssl-fips` package is available on the `ol6_x86_64_addons` channel.

To make an Oracle Linux Release 6 Update 5 or later system compliant with Federal Information Processing Standard (FIPS) Publication 140-2, perform the following steps:

1. Log in to ULN and enable the `ol6_x86_64_addons` channel for your system.

2. Remove the existing `openssl` package and install the `openssl-fips-1.0.1*` package. You can use `yum shell` to perform these transactions as shown here:

   ```
   # yum -y shell <<EOF
   remove openssl
   install openssl-fips-1.0.1*
   run
   EOF
   ```

   You cannot use separate `yum remove` and `yum install` commands as `yum` itself depends on the OpenSSL library being available.

   Alternatively, download the `openssl-fips-1.0.1*` package and use the `rpm` command instead:

   ```
   # rpm -e --nodeps openssl
   # rpm -ivh openssl-fips-1.0.1*.rpm
   ```

3. Identify either the device file path (*`device`*) under `/dev` of your system's boot device or its UUID (*`uuid`*) by using `ls -l` to examine the entries under `/dev/disk/by-uuid`, for example:

   ```
   # ls -l /dev/disk/by-uuid/
   total 0
   lrwxrwxrwx. 1 root root 10 Oct 27 09:37 873fd086-a84f-4ad4-878c-66287bda05bc -> ../../dm-1
   ```

```
lrwxrwxrwx. 1 root root 10 Oct 27 09:37 97bb06bb-fe5e-4d98-88b4-ba60bb4e7aec -> ../../dm-0
lrwxrwxrwx. 1 root root 10 Oct 27 09:38 ad8113d7-b279-4da8-b6e4-cfba045f66ff -> ../../sda1
```

In this example, the partition that contains `/boot` is `/dev/sda1`, which has the UUID `ad8113d7-b279-4da8-b6e4-cfba045f66ff`.

4. Edit `/etc/grub.conf`:

   a. Add a `fips=1` entry to the `kernel` command line to enable strict FIPS compliance.

   b. Add either a `boot=`*`device`* entry or a `boot=UUID=`*`uuid`* entry for the boot device to the `kernel` command line, for example:

   ```
   boot=UUID=ad8113d7-b279-4da8-b6e4-cfba045f66ff
   ```

   Caution

   If `/boot` or `/boot/efi` exist on a separate partition from `/` and you do not specify a valid `boot=` entry, the system crashes because it cannot locate the kernel's `.hmac` file.

5. Edit `/etc/sysconfig/prelink` and set `PRELINKING=no`.

   Prelinking must be disabled to allow the system to verify the integrity of modules correctly.

6. Remove all existing prelinking from binaries and libraries:

   ```
   # prelink -u -a
   ```

   Caution

   If you do not disable and remove all prelinking, users cannot log in and `/usr/sbin/sshd` does not start.

7. Install the `dracut-fips` package:

   ```
   # yum install dracut-fips
   ```

   This package must be installed so that the system checks the integrity of the kernel components at boot time.

8. Recreate the `initramfs` file system:

   ```
   # dracut -f
   ```

9. Remove the existing SSH host keys:

   ```
   # rm /etc/ssh/ssh_host*
   ```

   OpenSSH uses the FIPS-validated OpenSSL library modules to generate new, FIPS-approved keys when the system is next rebooted. (Under FIPS mode, `ssh-keygen` can create new RSA host keys in `/etc/ssh`, but not DSA keys, and it displays key fingerprints as SHA1 hashes instead of as MD5 hashes.)

10. Shut down and reboot the system into FIPS mode.

Note

While the system is rebooting, generate input events by pressing keys at random or by moving the mouse. You should create at least 256 such events to ensure that the system has sufficient entropy available for key generation.

6.3 Using the FIPS-Compliant OpenSSL Library

If the kernel command line specifies a `fips=1` entry, the value of `/proc/sys/crypto/fips_enabled` is set to 1, which causes the OpenSSL library module to initialize the FIPS-approved mode of operation automatically. To handle automatic initialization, an application that uses the module must call one of the following routines:

`void OPENSSL_add_all_algorithms(void)`	Calls `OPENSSL_init()` implicitly and adds all approved algorithms to the EVP API in FIPS-approved mode.
`void OPENSSL_init_library(void)`	Performs basic initialization of the library and initialize FIPS-approved mode without setting up the EVP API with supported algorithms.
`void SSL_library_init(void)`	Calls `OPENSSL_init()` implicitly, adds algorithms that are necessary for TLS protocol support and initializes the SSL library.

To put the library into FIPS-approved mode explicitly, an application can call the `int FIPS_mode_set(int on)` function. If the value of *on* is set to 1, the library switches from non-approved to approved mode. If any self tests or integrity verification tests fail, the library is put into the error state and the function returns 0. If the tests succeed, the function returns 1. If the value of *on* is set to 0, the library switches to non-approved mode. Alternatively, the application can call `OPENSSL_conf(const char *config_name)` to enable FIPS mode by reading the `alg_section` that is defined for the *config_name* entry in the standard configuration file (`openssl.conf`), for example:

```
[ config_name ]
alg_section = algsec
...
[ algsec ]
fips_mode = yes
```

`OPENSSL_config()` does not return a value. If there is an error in the configuration, the function writes a message to the standard error and forces the application to exit. To provide better error control, an application can call the `CONF_modules_load_file()` function instead.

An application can use the following functions to query the OpenSSL library module:

`int FIPS_mode(void)`	Returns 1 if the module is in FIPS-approved mode; otherwise it returns 0.
`int FIPS_selftest_failed(void)`	Returns 1 if the module is in the error state; otherwise it returns 0.

To set the FIPS random number generator key and internal state to zero, an application can call the `void RAND_cleanup(void)` function.

Note

If you set the value of the `OPENSSL_FIPS` environment variable to 1, the `openssl` binary that is included in the `openssl-fips-1.0.1*` package, and which has been built using the FIPS-compliant OpenSSL library, uses only FIPS 140-2

> approved algorithms. The value of `OPENSSL_FIPS` has no effect on the FIPS mode of the system. Do not assume that the value of `OPENSSL_FIPS` has any effect on other applications that use the FIPS-compliant OpenSSL library.

For more information about using the OpenSSL library with FIPS, see http://www.openssl.org/docs/fips/UserGuide-2.0.pdf.