Red Hat Enterprise Linux 9

Configuring basic system settings

A guide to configuring basic system settings in Red Hat Enterprise Linux 9
Abstract

This document describes basics of system administration on Red Hat Enterprise Linux 9. The title focuses on: basic tasks that a system administrator needs to do just after the operating system has been successfully installed, installing software with DNF, using systemd for service management, managing users, groups and file permissions, using chrony to configure NTP and others.
# Table of Contents

**MAKING OPEN SOURCE MORE INCLUSIVE**                                  7

**PROVIDING FEEDBACK ON RED HAT DOCUMENTATION**                        8

**CHAPTER 1. PREPARING A CONTROL NODE AND MANAGED NODES TO USE RHEL SYSTEM ROLES**  9  
1.1. INTRODUCTION TO RHEL SYSTEM ROLES                                  9  
1.2. RHEL SYSTEM ROLES TERMINOLOGY                                      9  
1.3. PREPARING A CONTROL NODE                                           10  
1.4. PREPARING A MANAGED NODE                                            12  
1.5. VERIFYING ACCESS FROM THE CONTROL NODE TO MANAGED NODE             13

**CHAPTER 2. CHANGING BASIC ENVIRONMENT SETTINGS**                     15  
2.1. CONFIGURING THE DATE AND TIME                                       15  
2.1.1. Displaying the current date and time                             15  
2.2. CONFIGURING THE SYSTEM LOCALE                                      15  
2.3. CONFIGURING THE KEYBOARD LAYOUT                                    16  
2.4. CHANGING THE LANGUAGE USING DESKTOP GUI                            17  
2.5. ADDITIONAL RESOURCES                                               19

**CHAPTER 3. CONFIGURING AND MANAGING NETWORK ACCESS**                 20  
3.1. CONFIGURING THE NETWORK AND HOST NAME IN THE GRAPHICAL INSTALLATION MODE 20  
3.2. CONFIGURING A STATIC ETHERNET CONNECTION USING NMCLI                21  
3.3. CONFIGURING A DYNAMIC ETHERNET CONNECTION USING NMTUI               23  
3.4. CONFIGURING A STATIC ETHERNET CONNECTION USING NMTUI                24  
3.5. MANAGING NETWORKING IN THE RHEL WEB CONSOLE                         27  
3.6. MANAGING NETWORKING USING RHEL SYSTEM ROLES                         28  
3.7. ADDITIONAL RESOURCES                                               29

**CHAPTER 4. REGISTERING THE SYSTEM AND MANAGING SUBSCRIPTIONS**       30  
4.1. REGISTERING THE SYSTEM AFTER THE INSTALLATION                       30  
4.2. REGISTERING SUBSCRIPTIONS WITH CREDENTIALS IN THE WEB CONSOLE       31  
4.3. REGISTERING A SYSTEM USING RED HAT ACCOUNT ON GNOME                 32  
4.4. REGISTERING A SYSTEM USING AN ACTIVATION KEY ON GNOME               33

**CHAPTER 5. MAKING SYSTEMD SERVICES START AT BOOT TIME**             35  
5.1. ENABLING OR DISABLING SERVICES                                     35

**CHAPTER 6. CONFIGURING SYSTEM SECURITY**                             36  
6.1. ENABLING THE FIREWALLD SERVICE                                      36  
6.2. MANAGING BASIC SELINUX SETTINGS                                    37  
6.3. ENSURING THE REQUIRED STATE OF SELINUX                             37  
6.4. ADDITIONAL RESOURCES                                               38

**CHAPTER 7. GETTING STARTED WITH MANAGING USER ACCOUNTS**            39  
7.1. MANAGING ACCOUNTS AND GROUPS USING COMMAND LINE TOOLS             39  
7.2. SYSTEM USER ACCOUNTS MANAGED IN THE WEB CONSOLE                    40  
7.3. ADDING NEW ACCOUNTS USING THE WEB CONSOLE                           40

**CHAPTER 8. DUMPING A CRASHED KERNEL FOR LATER ANALYSIS**           42  
8.1. WHAT IS KDUMP                                                       42  
8.2. CONFIGURING KDUMP MEMORY USAGE AND TARGET LOCATION IN WEB CONSOLE 42  
8.3. KDUMP USING RHEL SYSTEM ROLES                                      44  
8.4. ADDITIONAL RESOURCES                                               45
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>Recovering and Restoring a System</td>
<td>46</td>
</tr>
<tr>
<td>9.1</td>
<td>Setting Up REAR</td>
<td>46</td>
</tr>
<tr>
<td>9.2</td>
<td>Using a REAR Rescue Image on the 64-Bit IBM Z Architecture</td>
<td>47</td>
</tr>
<tr>
<td>10</td>
<td>Troubleshooting Problems Using Log Files</td>
<td>50</td>
</tr>
<tr>
<td>10.1</td>
<td>Services Handling Syslog Messages</td>
<td>50</td>
</tr>
<tr>
<td>10.2</td>
<td>Subdirectories Storing Syslog Messages</td>
<td>50</td>
</tr>
<tr>
<td>10.3</td>
<td>Inspecting Log Files Using the Web Console</td>
<td>50</td>
</tr>
<tr>
<td>10.4</td>
<td>Viewing Logs Using the Command Line</td>
<td>51</td>
</tr>
<tr>
<td>10.5</td>
<td>Additional Resources</td>
<td>52</td>
</tr>
<tr>
<td>11</td>
<td>Accessing the Red Hat Support</td>
<td>53</td>
</tr>
<tr>
<td>11.1</td>
<td>Obtaining Red Hat Support Through Red Hat Customer Portal</td>
<td>53</td>
</tr>
<tr>
<td>11.2</td>
<td>Troubleshooting Problems Using SOSREPORT</td>
<td>53</td>
</tr>
<tr>
<td>12</td>
<td>Introduction to Systemd</td>
<td>55</td>
</tr>
<tr>
<td>13</td>
<td>Managing System Services with Systemctl</td>
<td>57</td>
</tr>
<tr>
<td>13.1</td>
<td>Listing System Services</td>
<td>57</td>
</tr>
<tr>
<td>13.2</td>
<td>Displaying System Service Status</td>
<td>58</td>
</tr>
<tr>
<td>13.3</td>
<td>Starting a System Service</td>
<td>60</td>
</tr>
<tr>
<td>13.4</td>
<td>Stopping a System Service</td>
<td>61</td>
</tr>
<tr>
<td>13.5</td>
<td>Restarting a System Service</td>
<td>62</td>
</tr>
<tr>
<td>13.6</td>
<td>Enabling a System Service</td>
<td>63</td>
</tr>
<tr>
<td>13.7</td>
<td>Disabling a System Service</td>
<td>64</td>
</tr>
<tr>
<td>14</td>
<td>Working with Systemd Targets</td>
<td>65</td>
</tr>
<tr>
<td>14.1</td>
<td>Viewing the Default Target</td>
<td>65</td>
</tr>
<tr>
<td>14.2</td>
<td>Viewing the Target Units</td>
<td>65</td>
</tr>
<tr>
<td>14.3</td>
<td>Changing the Default Target</td>
<td>66</td>
</tr>
<tr>
<td>14.4</td>
<td>Changing the Default Target Using a Symbolic Link</td>
<td>67</td>
</tr>
<tr>
<td>14.5</td>
<td>Changing the Current Target</td>
<td>68</td>
</tr>
<tr>
<td>14.6</td>
<td>Booting to Rescue Mode</td>
<td>68</td>
</tr>
<tr>
<td>14.7</td>
<td>Booting to Emergency Mode</td>
<td>69</td>
</tr>
<tr>
<td>15</td>
<td>Shutting Down, Suspending, and Hibernating the System</td>
<td>70</td>
</tr>
<tr>
<td>15.1</td>
<td>System Shutdown</td>
<td>70</td>
</tr>
<tr>
<td>15.2</td>
<td>Shutting Down the System Using the Shutdown Command</td>
<td>70</td>
</tr>
<tr>
<td>15.3</td>
<td>Shutting Down the System Using the Systemctl Command</td>
<td>71</td>
</tr>
<tr>
<td>15.4</td>
<td>Restarting the System</td>
<td>71</td>
</tr>
<tr>
<td>15.5</td>
<td>Suspending the System</td>
<td>72</td>
</tr>
<tr>
<td>15.6</td>
<td>Hibernating the System</td>
<td>72</td>
</tr>
<tr>
<td>15.7</td>
<td>Overview of the Power Management Commands with Systemctl</td>
<td>73</td>
</tr>
<tr>
<td>16</td>
<td>Working with Systemd Unit Files</td>
<td>74</td>
</tr>
<tr>
<td>16.1</td>
<td>Introduction to Unit Files</td>
<td>74</td>
</tr>
<tr>
<td>16.2</td>
<td>Unit File Structure</td>
<td>74</td>
</tr>
<tr>
<td>16.3</td>
<td>Important [Unit] Section Options</td>
<td>75</td>
</tr>
<tr>
<td>16.4</td>
<td>Important [Service] Section Options</td>
<td>76</td>
</tr>
<tr>
<td>16.5</td>
<td>Important [Install] Section Options</td>
<td>77</td>
</tr>
<tr>
<td>16.6</td>
<td>Creating Custom Unit Files</td>
<td>77</td>
</tr>
<tr>
<td>16.7</td>
<td>Creating a Custom Unit File by Using the Second Instance of the SSHD Service</td>
<td>79</td>
</tr>
<tr>
<td>16.8</td>
<td>Converting Sysv Init Scripts to Unit Files</td>
<td>80</td>
</tr>
<tr>
<td>16.9</td>
<td>Finding the Systemd Service Description</td>
<td>81</td>
</tr>
<tr>
<td>16.10</td>
<td>Finding the Systemd Service Dependencies</td>
<td>81</td>
</tr>
<tr>
<td>Chapter</td>
<td>Title</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>16.11</td>
<td>FINDING DEFAULT TARGETS OF THE SERVICE</td>
<td>82</td>
</tr>
<tr>
<td>16.12</td>
<td>FINDING FILES USED BY THE SERVICE</td>
<td>82</td>
</tr>
<tr>
<td>16.13</td>
<td>MODIFYING EXISTING UNIT FILES</td>
<td>84</td>
</tr>
<tr>
<td>16.14</td>
<td>EXTENDING THE DEFAULT UNIT CONFIGURATION</td>
<td>85</td>
</tr>
<tr>
<td>16.15</td>
<td>OVERRIDING THE DEFAULT UNIT CONFIGURATION</td>
<td>86</td>
</tr>
<tr>
<td>16.16</td>
<td>CHANGING THE TIMEOUT LIMIT</td>
<td>87</td>
</tr>
<tr>
<td>16.17</td>
<td>MONITORING OVERRIDDEN UNITS</td>
<td>87</td>
</tr>
<tr>
<td>16.18</td>
<td>WORKING WITH INSTANTIATED UNITS</td>
<td>88</td>
</tr>
<tr>
<td>16.19</td>
<td>IMPORTANT UNIT SPECIFIERS</td>
<td>89</td>
</tr>
<tr>
<td>16.20</td>
<td>ADDITIONAL RESOURCES</td>
<td>90</td>
</tr>
<tr>
<td></td>
<td><strong>CHAPTER 17. OPTIMIZING SYSTEMD TO SHORTEN THE BOOT TIME</strong></td>
<td>91</td>
</tr>
<tr>
<td>17.1</td>
<td>EXAMINING SYSTEM BOOT PERFORMANCE</td>
<td>91</td>
</tr>
<tr>
<td></td>
<td>Analyzing overall boot time</td>
<td>91</td>
</tr>
<tr>
<td></td>
<td>Analyzing unit initialization time</td>
<td>91</td>
</tr>
<tr>
<td></td>
<td>Identifying critical units</td>
<td>91</td>
</tr>
<tr>
<td>17.2</td>
<td>A GUIDE TO SELECTING SERVICES THAT CAN BE SAFELY DISABLED</td>
<td>92</td>
</tr>
<tr>
<td>17.3</td>
<td>ADDITIONAL RESOURCES</td>
<td>96</td>
</tr>
<tr>
<td></td>
<td><strong>CHAPTER 18. INTRODUCTION TO MANAGING USER AND GROUP ACCOUNTS</strong></td>
<td>97</td>
</tr>
<tr>
<td>18.1</td>
<td>INTRODUCTION TO USERS AND GROUPS</td>
<td>97</td>
</tr>
<tr>
<td>18.2</td>
<td>CONFIGURING RESERVED USER AND GROUP IDS</td>
<td>97</td>
</tr>
<tr>
<td>18.3</td>
<td>USER PRIVATE GROUPS</td>
<td>98</td>
</tr>
<tr>
<td></td>
<td><strong>CHAPTER 19. MANAGING USER ACCOUNTS IN THE WEB CONSOLE</strong></td>
<td>99</td>
</tr>
<tr>
<td>19.1</td>
<td>SYSTEM USER ACCOUNTS MANAGED IN THE WEB CONSOLE</td>
<td>99</td>
</tr>
<tr>
<td>19.2</td>
<td>ADDING NEW ACCOUNTS USING THE WEB CONSOLE</td>
<td>99</td>
</tr>
<tr>
<td>19.3</td>
<td>ENFORCING PASSWORD EXPIRATION IN THE WEB CONSOLE</td>
<td>100</td>
</tr>
<tr>
<td>19.4</td>
<td>TERMINATING USER SESSIONS IN THE WEB CONSOLE</td>
<td>101</td>
</tr>
<tr>
<td></td>
<td><strong>CHAPTER 20. MANAGING USERS FROM THE COMMAND LINE</strong></td>
<td>103</td>
</tr>
<tr>
<td>20.1</td>
<td>ADDING A NEW USER FROM THE COMMAND LINE</td>
<td>103</td>
</tr>
<tr>
<td>20.2</td>
<td>ADDING A NEW GROUP FROM THE COMMAND LINE</td>
<td>103</td>
</tr>
<tr>
<td>20.3</td>
<td>ADDING A USER TO A SUPPLEMENTARY GROUP FROM THE COMMAND LINE</td>
<td>104</td>
</tr>
<tr>
<td>20.4</td>
<td>CREATING A GROUP DIRECTORY</td>
<td>105</td>
</tr>
<tr>
<td></td>
<td><strong>CHAPTER 21. EDITING USER GROUPS USING THE COMMAND LINE</strong></td>
<td>107</td>
</tr>
<tr>
<td>21.1</td>
<td>PRIMARY AND SUPPLEMENTARY USER GROUPS</td>
<td>107</td>
</tr>
<tr>
<td>21.2</td>
<td>LISTING THE PRIMARY AND SUPPLEMENTARY GROUPS OF A USER</td>
<td>107</td>
</tr>
<tr>
<td>21.3</td>
<td>CHANGING THE PRIMARY GROUP OF A USER</td>
<td>108</td>
</tr>
<tr>
<td>21.4</td>
<td>ADDING A USER TO A SUPPLEMENTARY GROUP FROM THE COMMAND LINE</td>
<td>109</td>
</tr>
<tr>
<td>21.5</td>
<td>REMOVING A USER FROM A SUPPLEMENTARY GROUP</td>
<td>109</td>
</tr>
<tr>
<td>21.6</td>
<td>CHANGING ALL OF THE SUPPLEMENTARY GROUPS OF A USER</td>
<td>110</td>
</tr>
<tr>
<td></td>
<td><strong>CHAPTER 22. MANAGING SUDO ACCESS</strong></td>
<td>112</td>
</tr>
<tr>
<td>22.1</td>
<td>USER AUTHORIZATIONS IN SUDOERS</td>
<td>112</td>
</tr>
<tr>
<td>22.2</td>
<td>GRANTING SUDO ACCESS TO A USER</td>
<td>113</td>
</tr>
<tr>
<td>22.3</td>
<td>ENABLING UNPRIVILEGED USERS TO RUN CERTAIN COMMANDS</td>
<td>114</td>
</tr>
<tr>
<td></td>
<td><strong>CHAPTER 23. CHANGING AND RESETTING THE ROOT PASSWORD</strong></td>
<td>116</td>
</tr>
<tr>
<td>23.1</td>
<td>CHANGING THE ROOT PASSWORD AS THE ROOT USER</td>
<td>116</td>
</tr>
<tr>
<td>23.2</td>
<td>CHANGING OR RESETTING THE FORGOTTEN ROOT PASSWORD AS A NON-ROOT USER</td>
<td>116</td>
</tr>
<tr>
<td>23.3</td>
<td>RESETTING THE ROOT PASSWORD ON BOOT</td>
<td>116</td>
</tr>
<tr>
<td></td>
<td><strong>CHAPTER 24. MANAGING FILE PERMISSIONS</strong></td>
<td>119</td>
</tr>
</tbody>
</table>
# 24. BASE FILE PERMISSIONS

- **24.1. BASE FILE PERMISSIONS**
- **24.2. USER FILE-CREATION MODE MASK**
- **24.3. DEFAULT FILE PERMISSIONS**
- **24.4. CHANGING FILE PERMISSIONS USING SYMBOLIC VALUES**
- **24.5. CHANGING FILE PERMISSIONS USING OCTAL VALUES**

# 25. MANAGING THE UMASK

- **25.1. DISPLAYING THE CURRENT VALUE OF THE UMASK**
- **25.2. DISPLAYING THE DEFAULT BASH UMASK**
- **25.3. SETTING THE UMASK USING SYMBOLIC VALUES**
- **25.4. SETTING THE UMASK USING OCTAL VALUES**
- **25.5. CHANGING THE DEFAULT UMASK FOR THE NON-LOGIN SHELL**
- **25.6. CHANGING THE DEFAULT UMASK FOR THE LOGIN SHELL**
- **25.7. CHANGING THE DEFAULT UMASK FOR A SPECIFIC USER**
- **25.8. SETTING DEFAULT PERMISSIONS FOR NEWLY CREATED HOME DIRECTORIES**

# 26. MANAGING THE ACCESS CONTROL LIST

- **26.1. DISPLAYING THE CURRENT ACCESS CONTROL LIST**
- **26.2. SETTING THE ACCESS CONTROL LIST**

# 27. USING THE CHRONY SUITE TO CONFIGURE NTP

- **27.1. INTRODUCTION TO CHRONY SUITE**
- **27.2. USING CHRONYC TO CONTROL CHRONYD**

# 28. USING CHRONY

- **28.1. MANAGING CHRONY**
- **28.2. CHECKING IF CHRONY IS SYNCHRONIZED**
- **28.3. MANUALLY ADJUSTING THE SYSTEM CLOCK**
- **28.4. SETTING UP CHRONY FOR A SYSTEM IN AN ISOLATED NETWORK**
- **28.5. CONFIGURING REMOTE MONITORING ACCESS**
- **28.6. MANAGING TIME SYNCHRONIZATION USING RHEL SYSTEM ROLES**
- **28.7. ADDITIONAL RESOURCES**

# 29. CHRONY WITH HW TIMESTAMPING

- **29.1. VERIFYING SUPPORT FOR HARDWARE TIMESTAMPING**
- **29.2. ENABLING HARDWARE TIMESTAMPING**
- **29.3. CONFIGURING CLIENT POLLING INTERVAL**
- **29.4. ENABLING INTERLEAVED MODE**
- **29.5. CONFIGURING SERVER FOR LARGE NUMBER OF CLIENTS**
- **29.6. VERIFYING HARDWARE TIMESTAMPING**
- **29.7. CONFIGURING PTP-NTP BRIDGE**

# 30. OVERVIEW OF NETWORK TIME SECURITY (NTS) IN CHRONY

- **30.1. ENABLING NETWORK TIME SECURITY (NTS) IN THE CLIENT CONFIGURATION FILE**
- **30.2. ENABLING NETWORK TIME SECURITY (NTS) ON THE SERVER**

# 31. USING SECURE COMMUNICATIONS BETWEEN TWO SYSTEMS WITH OPENSSH

- **31.1. SSH AND OPENSSH**
- **31.2. CONFIGURING AND STARTING AN OPENSSH SERVER**
- **31.3. SETTING AN OPENSSH SERVER FOR KEY-BASED AUTHENTICATION**
- **31.4. GENERATING SSH KEY PAIRS**
- **31.5. USING SSH KEYS STORED ON A SMART CARD**
- **31.6. MAKING OPENSSH MORE SECURE**
- **31.7. CONNECTING TO A REMOTE SERVER USING AN SSH JUMP HOST**
- **31.8. CONNECTING TO REMOTE MACHINES WITH SSH KEYS USING SSH-AGENT**
Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our CTO Chris Wright’s message.
PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your feedback on our documentation. Let us know how we can improve it.

Submitting comments on specific passages

1. View the documentation in the Multi-page HTML format and ensure that you see the Feedback button in the upper right corner after the page fully loads.
2. Use your cursor to highlight the part of the text that you want to comment on.
3. Click the Add Feedback button that appears near the highlighted text.
4. Add your feedback and click Submit.

Submitting feedback through Bugzilla (account required)

1. Log in to the Bugzilla website.
2. Select the correct version from the Version menu.
3. Enter a descriptive title in the Summary field.
4. Enter your suggestion for improvement in the Description field. Include links to the relevant parts of the documentation.
5. Click Submit Bug.
CHAPTER 1. PREPARING A CONTROL NODE AND MANAGED NODES TO USE RHEL SYSTEM ROLES

Before you can use individual RHEL System Roles to manage services and settings, prepare the involved hosts.

1.1. INTRODUCTION TO RHEL SYSTEM ROLES

RHEL System Roles is a collection of Ansible roles and modules. RHEL System Roles provide a configuration interface to remotely manage multiple RHEL systems. The interface enables managing system configurations across multiple versions of RHEL, as well as adopting new major releases.

On Red Hat Enterprise Linux 9, the interface currently consists of the following roles:

- Certificate Issuance and Renewal
- Kernel Settings (kernel_settings)
- Metrics (metrics)
- Network Bound Disk Encryption client and Network Bound Disk Encryption server (nbde_client and nbde_server)
- Networking (network)
- Postfix (postfix)
- SSH client (ssh)
- SSH server (sshd)
- System-wide Cryptographic Policies (crypto_policies)
- Terminal Session Recording (tlog)

All these roles are provided by the rhel-system-roles package available in the AppStream repository.

Additional resources

- Red Hat Enterprise Linux (RHEL) System Roles
- /usr/share/doc/rHEL-system-roles/ provided by the rhel-system-roles package.

1.2. RHEL SYSTEM ROLES TERMINOLOGY

You can find the following terms across this documentation:

Ansible playbook

Playbooks are Ansible’s configuration, deployment, and orchestration language. They can describe a policy you want your remote systems to enforce, or a set of steps in a general IT process.

Control node

Any machine with Ansible installed. You can run commands and playbooks, invoking /usr/bin/ansible or /usr/bin/ansible-playbook, from any control node. You can use any computer that has Python installed on it as a control node - laptops, shared desktops, and servers can all run Ansible. However,
you cannot use a Windows machine as a control node. You can have multiple control nodes.

Inventory
A list of managed nodes. An inventory file is also sometimes called a “hostfile”. Your inventory can specify information like IP address for each managed node. An inventory can also organize managed nodes, creating and nesting groups for easier scaling. To learn more about inventory, see the Working with Inventory section.

Managed nodes
The network devices, servers, or both that you manage with Ansible. Managed nodes are also sometimes called “hosts”. Ansible is not installed on managed nodes.

1.3. PREPARING A CONTROL NODE

RHEL includes Ansible Core in the AppStream repository with a limited scope of support. If you require additional support for Ansible, contact Red Hat to learn more about the Ansible Automation Platform subscription.

Prerequisites
- You registered the system to the Customer Portal.
- You attached a Red Hat Enterprise Linux Server subscription to the system.
- If available in your Customer Portal account, you attached an Ansible Automation Platform subscription to the system.

Procedure
1. Install the rhel-system-roles package:
   ```bash
   [root@control-node]# dnf install rhel-system-roles
   ```
   This command installs Ansible Core as a dependency.

2. Create a user that you later use to manage and execute playbooks:
   ```bash
   [root@control-node]# useradd ansible
   ```

3. Switch to the newly created ansible user:
   ```bash
   [root@control-node]# su - ansible
   ```
   Perform the rest of the procedure as this user.

4. Create an SSH public and private key
   ```bash
   [ansible@control-node]$ ssh-keygen
   Generating public/private rsa key pair. Enter file in which to save the key (/home/ansible/.ssh/id_rsa): password ...
   ```
   Use the suggested default location for the key file.
5. Optional: Configure an SSH agent to prevent Ansible from prompting you for the SSH key password each time you establish a connection.

6. Create the `~/.ansible.cfg` file with the following content:

```ini
[defaults]
inventory = /home/ansible/inventory
remote_user = ansible

[privilege_escalation]
become = True
become_method = sudo
become_user = root
become_ask_pass = True
```

With these settings:

- Ansible manages hosts in the specified inventory file.

- Ansible uses the account set in the `remote_user` parameter when it establishes SSH connections to managed nodes.

- Ansible uses the `sudo` utility to execute tasks on managed nodes as the `root` user. For security reasons, configure `sudo` on managed nodes to require entering the password of the remote user to become `root`. By specifying the `become_ask_pass=True` setting in `~/.ansible.cfg`, Ansible prompts for this password when you execute a playbook.

Settings in the `~/.ansible.cfg` file have a higher priority and override settings from the global `/etc/ansible/ansible.cfg` file.

7. Create the `~/inventory` file. For example, the following is an inventory file in the INI format with three hosts and one host group named `US`:

```ini
managed-node-01.example.com

[US]
managed-node-02.example.com ansible_host=192.0.2.100
managed-node-03.example.com
```

Note that the control node must be able to resolve the hostnames. If the DNS server cannot resolve certain hostnames, add the `ansible_host` parameter next to the host entry to specify its IP address.

**Verification**

1. Prepare a managed node.

2. Verify access from the control node to managed nodes

**Additional resources**

- Scope of support for the Ansible Core package included in the RHEL 9 and RHEL 8.6 and later AppStream repositories

- How to register and subscribe a system to the Red Hat Customer Portal using subscription-manager
1.4. PREPARING A MANAGED NODE

Ansible does not use an agent on managed hosts. The only requirements are Python, which is installed by default on RHEL, and SSH access to the managed host.

However, direct SSH access as the root user can be a security risk. Therefore, when you prepare a managed node, you create a local user on this node and configure a sudo policy. Ansible on the control node can then use this account to log in to the managed node and execute playbooks as different users, such as root.

Prerequisites

- You prepared the control node.

Procedure

1. Create a user:

   [root@managed-node-01]# useradd ansible

   The control node later uses this user to establish an SSH connection to this host.

2. Set a password to the ansible user:

   [root@managed-node-01]# passwd ansible
   Changing password for user ansible.
   New password: password
   Retype new password: password
   passwd: all authentication tokens updated successfully.

   You must enter this password when Ansible uses sudo to perform tasks as the root user.

3. Install the ansible user’s SSH public key on the managed node:

   a. Log into the control node as the ansible user, and copy the SSH public key to the managed node:

      [ansible@control-node]# ssh-copy-id managed-node-01.example.com
      /usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed:
      "~/home/ansible/.ssh/id_rsa.pub"
      The authenticity of host 'managed-node-01.example.com (192.0.2.100)' can't be established.
      ECDSA key fingerprint is
      SHA256:9bZ33GJNODK3zbNhybokN/6Mq7hu3vpBXDrCxe7NAvo.
      Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
      /usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
ansible@managed-node-01.example.com's password: password

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'managed-node-01.example.com"
and check to make sure that only the key(s) you wanted were added.

b. Remotely execute a command on the control node to verify the SSH connection:

[ansible@control-node]$ ssh managed-node-01.example.com whoami
ansible

4. Create a sudo configuration for the ansible user:

a. Use the visudo command to create and edit the /etc/sudoers.d/ansible file:

[root@managed-node-01]# visudo /etc/sudoers.d/ansible

The benefit of using visudo over a normal editor is that this utility provides basic sanity checks and checks for parse errors before installing the file.

b. Enter the following content to the /etc/sudoers.d/ansible file:

ansible ALL=(ALL) NOPASSWD: ALL

These settings grant permissions to the ansible user to run all commands as any user and group on this host without entering the password of the ansible user.

Additional resources

- Preparing the control node
- The sudoers(5) man page

1.5. VERIFYING ACCESS FROM THE CONTROL NODE TO MANAGED NODES

After you configured the control node and prepared managed nodes, test that Ansible can connect to the managed nodes.

Perform this procedure as the ansible user on the control node.

Prerequisites

- You prepared the control node as described in Preparing a control node.
- You prepared at least one managed node as described in Preparing a managed node.
- If you want to run playbooks on host groups, the managed node is listed in the inventory file on the control node.

Procedure
1. Use the Ansible `ping` module to verify that you can execute commands on an all managed hosts:

```
[ansible@control-node]$ ansible all -m ping
BECOME password: password
managed-node-01.example.com | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "ping": "pong"
}
...
```

The hard-coded `all` host group dynamically contains all hosts listed in the inventory file.

2. Use the Ansible `command` module to run the `whoami` utility on a managed host:

```
[ansible@control-node]$ ansible managed-node-01.example.com -m command -a whoami
BECOME password: password
managed-node-01.example.com | CHANGED | rc=0 >>
root
```

If the command returns `root`, you configured `sudo` on the managed nodes correctly, and privilege escalation works.
CHAPTER 2. CHANGING BASIC ENVIRONMENT SETTINGS

Configuration of basic environment settings is a part of the installation process. The following sections guide you when you change them later. The basic configuration of the environment includes:

- Date and time
- System locales
- Keyboard layout
- Language

2.1. CONFIGURING THE DATE AND TIME

Accurate timekeeping is important for a number of reasons. In Red Hat Enterprise Linux, timekeeping is ensured by the NTP protocol, which is implemented by a daemon running in user space. The user-space daemon updates the system clock running in the kernel. The system clock can keep time by using various clock sources.

Red Hat Enterprise Linux 8 and later versions use the chronyd daemon to implement NTP. chronyd is available from the chrony package. For more information, see Using the chrony suite to configure NTP.

2.1.1. Displaying the current date and time

To display the current date and time, use either of these steps.

Procedure

1. Enter the `date` command:

   ```
   $ date
   Mon Mar 30 16:02:59 CEST 2020
   ```

2. To see more details, use the `timedatectl` command:

   ```
   $ timedatectl
   Local time: Mon 2020-03-30 16:04:42 CEST
   Universal time: Mon 2020-03-30 14:04:42 UTC
   RTC time: Mon 2020-03-30 14:04:41
   Time zone: Europe/Prague (CEST, +0200)
   System clock synchronized: yes
   NTP service: active
   RTC in local TZ: no
   ```

Additional resources

- Configuring time settings using the web console
- `man date(1)` and `man timedatectl(1)`

2.2. CONFIGURING THE SYSTEM LOCALE
System-wide locale settings are stored in the `/etc/locale.conf` file, which is read at early boot by the `systemd` daemon. Every service or user inherits the locale settings configured in `/etc/locale.conf`, unless individual programs or individual users override them.

This section describes how to manage system locale.

Procedure

- To list available system locale settings:
  ```
  $ localectl list-locales
  C.utf8
  aa_DJ
  aa_DJ.iso88591
  aa_DJ.utf8
  ...
  ```

- To display the current status of the system locales settings:
  ```
  $ localectl status
  ```

- To set or change the default system locale settings, use a `localectl set-locale` sub-command as the root user. For example:
  ```
  # localectl set-locale LANG=en_US
  ```

Additional resources

- man `localectl(1)`, `locale(7)`, and `locale.conf(5)`

2.3. CONFIGURING THE KEYBOARD LAYOUT

The keyboard layout settings control the layout used on the text console and graphical user interfaces.

Procedure

- To list available keymaps:
  ```
  $ localectl list-keymaps
  ANSI-dvorak
  al
  al-plisi
  amiga-de
  amiga-us
  ...
  ```

- To display the current status of keymaps settings:
  ```
  $ localectl status
  ...
  VC Keymap: us
  ```
To set or change the default system keymap. For example:

```
# localectl set-keymap us
```

**Additional resources**

- `man localectl(1)`, `man locale(7)`, and `man locale.conf(5)`

### 2.4. CHANGING THE LANGUAGE USING DESKTOP GUI

This section describes how to change the system language using the desktop GUI.

**Prerequisites**

- Required language packages are installed on your system

**Procedure**

1. Open the **GNOME Control Center** from the **System menu** by clicking on its icon.

2. In the **GNOME Control Center**, choose **Region & Language** from the left vertical bar.

3. Click the **Language** menu.

4. Select the required region and language from the menu.
If your region and language are not listed, scroll down, and click More to select from available regions and languages.

5. Click Done.

6. Click Restart for changes to take effect.
NOTE

Some applications do not support certain languages. The text of an application that cannot be translated into the selected language remains in US English.

Additional resources

- Launching applications in GNOME

2.5. ADDITIONAL RESOURCES

- Performing a standard RHEL 9 installation
CHAPTER 3. CONFIGURING AND MANAGING NETWORK ACCESS

This section describes different options on how to add Ethernet connections in Red Hat Enterprise Linux.

3.1. CONFIGURING THE NETWORK AND HOST NAME IN THE GRAPHICAL INSTALLATION MODE

Follow the steps in this procedure to configure your network and host name.

Procedure

1. From the Installation Summary window, click Network and Host Name.

2. From the list in the left-hand pane, select an interface. The details are displayed in the right-hand pane.

   NOTE
   - There are several types of network device naming standards used to identify network devices with persistent names, for example, `em1` and `wl3sp0`. For information about these standards, see the Configuring and managing networking document.

3. Toggle the ON/OFF switch to enable or disable the selected interface.

   NOTE
   - The installation program automatically detects locally accessible interfaces, and you cannot add or remove them manually.

4. Click + to add a virtual network interface, which can be either: Team (deprecated), Bond, Bridge, or VLAN.

5. Click - to remove a virtual interface.

6. Click Configure to change settings such as IP addresses, DNS servers, or routing configuration for an existing interface (both virtual and physical).

7. Type a host name for your system in the Host Name field.
NOTE

- The host name can be either a fully-qualified domain name (FQDN) in the format `hostname.domainname`, or a short host name with no domain name. Many networks have a Dynamic Host Configuration Protocol (DHCP) service that automatically supplies connected systems with a domain name. To allow the DHCP service to assign the domain name to this machine, specify only the short host name. The value `localhost` means that no specific static host name for the target system is configured, and the actual host name of the installed system is configured during the processing of the network configuration, for example, by `NetworkManager` using DHCP or DNS.

- Host names can only contain alpha-numeric characters and - or .. Host names cannot start or end with - and ..

8. Click **Apply** to apply the host name to the installer environment.

9. Alternatively, in the **Network and Hostname** window, you can choose the Wireless option. Click **Select network** in the right-hand pane to select your wifi connection, enter the password if required, and click **Done**.

Additional resources

- **Performing an advanced RHEL 9 installation**

### 3.2. CONFIGURING A STATIC ETHERNET CONNECTION USING NMCLI

This procedure describes adding an Ethernet connection with the following settings using the `nmcli` utility:

- A static IPv4 address - `192.0.2.1` with a `/24` subnet mask
- A static IPv6 address - `2001:db8:1::1` with a `/64` subnet mask
- An IPv4 default gateway - `192.0.2.254`
- An IPv6 default gateway - `2001:db8:1::fffe`
- An IPv4 DNS server - `192.0.2.200`
- An IPv6 DNS server - `2001:db8:1::ffbb`
- A DNS search domain - `example.com`

**Procedure**

1. Add a new NetworkManager connection profile for the Ethernet connection:

```bash
# nmcli connection add con-name Example-Connection ifname enp7s0 type ethernet
```

The further steps modify the **Example-Connection** connection profile you created.

2. Set the IPv4 address:

```bash
# nmcli connection modify Example-Connection ipv4.addresses 192.0.2.1/24
```
3. Set the IPv6 address:

   # nmcli connection modify Example-Connection ipv6.addresses 2001:db8:1::1/64

4. Set the IPv4 and IPv6 connection method to manual:

   # nmcli connection modify Example-Connection ipv4.method manual
   # nmcli connection modify Example-Connection ipv6.method manual

5. Set the IPv4 and IPv6 default gateways:

   # nmcli connection modify Example-Connection ipv4.gateway 192.0.2.254
   # nmcli connection modify Example-Connection ipv6.gateway 2001:db8:1::fffe

6. Set the IPv4 and IPv6 DNS server addresses:

   # nmcli connection modify Example-Connection ipv4.dns "192.0.2.200"
   # nmcli connection modify Example-Connection ipv6.dns "2001:db8:1::ffbb"

   To set multiple DNS servers, specify them space-separated and enclosed in quotes.

7. Set the DNS search domain for the IPv4 and IPv6 connection:

   # nmcli connection modify Example-Connection ipv4.dns-search example.com
   # nmcli connection modify Example-Connection ipv6.dns-search example.com

8. Activate the connection profile:

   # nmcli connection up Example-Connection
   Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/13)

Verification steps

1. Display the status of the devices and connections:

   # nmcli device status
   DEVICE   TYPE      STATE      CONNECTION
   enp7s0    ethernet  connected  Example-Connection

2. Use the ping utility to verify that this host can send packets to other hosts:

   # ping host_name_or_IP_address

Troubleshooting

- Make sure that the network cable is plugged-in to the host and a switch.

- Check whether the link failure exists only on this host or also on other hosts connected to the same switch.
Verify that the network cable and the network interface are working as expected. Perform hardware diagnosis steps and replace defect cables and network interface cards.

If the configuration on the disk does not match the configuration on the device, starting or restarting NetworkManager creates an in-memory connection that reflects the configuration of the device. For further details and how to avoid this problem, see NetworkManager duplicates a connection after restart of NetworkManager service.

Additional resources

- nm-settings(5) man page
- nmcli(1) man page
- Configuring NetworkManager to avoid using a specific profile to provide a default gateway

3.3. CONFIGURING A DYNAMIC ETHERNET CONNECTION USING NMTUI

The nmtui application provides a text-based user interface for NetworkManager. You can use nmtui to configure an Ethernet connection with a dynamic IP address on a host without a graphical interface.

NOTE

In nmtui:

- Navigate by using the cursor keys.
- Press a button by selecting it and hitting Enter.
- Select and deselect checkboxes by using Space.

Procedure

1. If you do not know the network device name you want to use in the connection, display the available devices:

   ```shell
   # nmcli device status
   DEVICE     TYPE      STATE                   CONNECTION
   enp7s0     ethernet  unavailable             --
   ...
   ```

2. Start nmtui:

   ```shell
   # nmtui
   ```

3. Select Edit a connection, and press Enter.

4. Press the Add button.

5. Select Ethernet from the list of network types, and press Enter.

6. Optional: Enter a name for the NetworkManager profile to be created.
7. Enter the network device name into the **Device** field.

8. Press the **OK** button to create and automatically activate the new connection.

![Edit Connection Screen]

9. Press the **Back** button to return to the main menu.

10. Select **Quit**, and press **Enter** to close the **nmtui** application.

**Verification**

1. Display the status of the devices and connections:

   ```
   # nmcli device status
   DEVICE      TYPE      STATE      CONNECTION
   enp7s0      ethernet  connected  Example-Connection
   ```

2. Use the **ping** utility to verify that this host can send packets to other hosts:

   ```
   # ping host_name_or_IP_address
   ```

**Troubleshooting**

- Make sure that the network cable is plugged-in to the host and a switch.

- Check whether the link failure exists only on this host or also on other hosts connected to the same switch.

- Verify that the network cable and the network interface are working as expected. Perform hardware diagnosis steps and replace defect cables and network interface cards.

- If the configuration on the disk does not match the configuration on the device, starting or restarting NetworkManager creates an in-memory connection that reflects the configuration of the device. For further details and how to avoid this problem, see NetworkManager duplicates a connection after restart of NetworkManager service.

**Additional resources**

- Configuring NetworkManager to avoid using a specific profile to provide a default gateway

**3.4. CONFIGURING A STATIC ETHERNET CONNECTION USING NMTUI**
The **nmtui** application provides a text-based user interface for NetworkManager. You can use **nmtui** to configure an Ethernet connection with a static IP address on a host without a graphical interface.

**NOTE**

In **nmtui**:

- Navigate by using the cursor keys.
- Press a button by selecting it and hitting **Enter**.
- Select and deselect checkboxes by using **Space**.

**Procedure**

1. If you do not know the network device name you want to use in the connection, display the available devices:

   ```
   # nmcli device status
   DEVICE     TYPE      STATE                   CONNECTION
   enp7s0      ethernet  unavailable             --
   ...
   ```

2. Start **nmtui**:

   ```
   # nmtui
   ```

3. Select **Edit a connection**, and press **Enter**.

4. Press the **Add** button.

5. Select **Ethernet** from the list of network types, and press **Enter**.

6. Optional: Enter a name for the NetworkManager profile to be created.

7. Enter the network device name into the **Device** field.

8. Configure the IPv4 and IPv6 address settings in the **IPv4 configuration** and **IPv6 configuration** areas:

   a. Press the **Automatic** button, and select **Manual** from the displayed list.

   b. Press the **Show** button next to the protocol you want to configure to display additional fields.

   c. Press the **Add** button next to **Addresses**, and enter the IP address and the subnet mask in Classless Inter-Domain Routing (CIDR) format.

   If you do not specify a subnet mask, NetworkManager sets a /**32** subnet mask for IPv4 addresses and /**64** for IPv6 addresses.

   d. Enter the address of the default gateway.

   e. Press the **Add** button next to **DNS servers**, and enter the DNS server address.

   f. Press the **Add** button next to **Search domains**, and enter the DNS search domain.
9. Press the **OK** button to create and automatically activate the new connection.

10. Press the **Back** button to return to the main menu.

11. Select **Quit**, and press **Enter** to close the `nmcli` application.

**Verification**

1. Display the status of the devices and connections:

```bash
# nmcli device status
DEVICE   TYPE    STATE     CONNECTION
enp7s0   ethernet connected Example-Connection
```
2. Use the **ping** utility to verify that this host can send packets to other hosts:

   ```
   # ping host_name_or_IP_address
   ```

**Troubleshooting**

- Make sure that the network cable is plugged-in to the host and a switch.
- Check whether the link failure exists only on this host or also on other hosts connected to the same switch.
- Verify that the network cable and the network interface are working as expected. Perform hardware diagnosis steps and replace defect cables and network interface cards.
- If the configuration on the disk does not match the configuration on the device, starting or restarting NetworkManager creates an in-memory connection that reflects the configuration of the device. For further details and how to avoid this problem, see NetworkManager duplicates a connection after restart of NetworkManager service.

**Additional resources**

- Configuring NetworkManager to avoid using a specific profile to provide a default gateway

### 3.5. MANAGING NETWORKING IN THE RHEL WEB CONSOLE

In the web console, the **Networking** menu enables you:

- To display currently received and sent packets
- To display the most important characteristics of available network interfaces
- To display content of the networking logs.
- To add various types of network interfaces (bond, team, bridge, VLAN)

**Figure 3.2. Managing Networking in the RHEL web console**
3.6. MANAGING NETWORKING USING RHEL SYSTEM ROLES

You can configure the networking connections on multiple target machines using the network role.

The network role allows to configure the following types of interfaces:

- Ethernet
- Bridge
- Bonded
- VLAN
- MacVLAN
- InfiniBand

The required networking connections for each host are provided as a list within the `network_connections` variable.

---

**WARNING**

The network role updates or creates all connection profiles on the target system exactly as specified in the `network_connections` variable. Therefore, the network role removes options from the specified profiles if the options are only present on the system but not in the `network_connections` variable.

---

The following example shows how to apply the `network` role to ensure that an Ethernet connection with the required parameters exists:

**An example playbook applying the network role to set up an Ethernet connection with the required parameters**

```yaml
# SPDX-License-Identifier: BSD-3-Clause
---
- hosts: managed-node-01.example.com
  vars:
    network_connections:

      # Create one Ethernet profile and activate it.
      # The profile uses automatic IP addressing
      # and is tied to the interface by MAC address.
      - name: prod1
        state: up
        type: ethernet
        autoconnect: yes
        mac: "00:00:5e:00:53:00"
        mtu: 1450
```

---
roles:
- rhel-system-roles.network

Additional resources

- Preparing a control node and managed nodes to use RHEL System Roles

3.7. ADDITIONAL RESOURCES

- Configuring and managing networking
CHAPTER 4. REGISTERING THE SYSTEM AND MANAGING SUBSCRIPTIONS

Subscriptions cover products installed on Red Hat Enterprise Linux, including the operating system itself.

You can use a subscription to Red Hat Content Delivery Network to track:

- Registered systems
- Products installed on your systems
- Subscriptions attached to the installed products

4.1. REGISTERING THE SYSTEM AFTER THE INSTALLATION

Use the following procedure to register your system if you have not registered it during the installation process already.

Prerequisites

- A valid user account in the Red Hat Customer Portal.
- See the Create a Red Hat Login page.
- An active subscription for the RHEL system.
- For more information about the installation process, see Performing a standard RHEL 9 installation.

Procedure

1. Register and automatically subscribe your system in one step:

   ```bash
   # subscription-manager register --username <username> --password <password> --auto-attach
   Registering to: subscription.rhsm.redhat.com:443/subscription
   The system has been registered with ID: 37to907c-ece6-49ea-9174-20b87ajk9ee7
   The registered system name is: client1.idm.example.com
   Installed Product Current Status:
   Product Name: Red Hat Enterprise Linux for x86_64
   Status: Subscribed
   
   The command prompts you to enter your Red Hat Customer Portal user name and password.
   
   If the registration process fails, you can register your system with a specific pool. For guidance on how to do it, proceed with the following steps:
   
   a. Determine the pool ID of a subscription that you require:

      ```bash
      # subscription-manager list --available
      
      This command displays all available subscriptions for your Red Hat account. For every subscription, various characteristics are displayed, including the pool ID.
b. Attach the appropriate subscription to your system by replacing `pool_id` with the pool ID determined in the previous step:

```
# subscription-manager attach --pool=pool_id
```

**NOTE**

To register the system with Red Hat Insights, you can use the `rhc connect` utility. See [Setting up Red Hat connector](#).

Additional resources

- Understanding autoattaching subscriptions on the Customer Portal
- Understanding the manual registration and subscription on the Customer Portal

### 4.2. REGISTERING SUBSCRIPTIONS WITH CREDENTIALS IN THE WEB CONSOLE

Use the following steps to register a newly installed Red Hat Enterprise Linux with account credentials using the RHEL web console.

**Prerequisites**

- A valid user account on the Red Hat Customer Portal. See the [Create a Red Hat Login](#) page.
- Active subscription for your RHEL system.

**Procedure**

1. Log in to the RHEL web console. For details, see [Logging in to the web console](#).

2. In the **Health** filed in the **Overview** page, click the **Not registered** warning, or click **Subscriptions** in the main menu to move to page with your subscription information.

3. In the **Overview** filed, click **Register**.
4. In the **Register system** dialog box, select that you want to register using your account credentials.

5. Enter your username.

6. Enter your password.

7. Optionally, enter your organization’s name or ID. If your account belongs to more than one organization on the Red Hat Customer Portal, you have to add the organization name or organization ID. To get the org ID, go to your Red Hat contact point.

   - If you do not want to connect your system to Red Hat Insights, clear the **Insights** check box.

8. Click the **Register** button.

At this point, your Red Hat Enterprise Linux system has been successfully registered.

### 4.3. REGISTERING A SYSTEM USING RED HAT ACCOUNT ON GNOME

Follow the steps in this procedure to enroll your system with your Red Hat account.

**Prerequisites**
• A valid account on Red Hat customer portal.
See the Create a Red Hat Login page for new user registration.

Procedure

1. Open the system menu, which is accessible from the upper-right screen corner, and click Settings.

2. Go to About → Subscription.

3. If you are not using the Red Hat server:
   a. In the Registration Server section, select Custom Address.
   b. Enter the server address in the URL field.

4. In the Registration Type section, select Red Hat Account.

5. In the Registration Details section:
   • Enter your Red Hat account user name in the Login field.
   • Enter your Red Hat account password in the Password field.
   • Enter the name of your organization in the Organization field.

6. Click Register.

4.4. REGISTERING A SYSTEM USING AN ACTIVATION KEY ON GNOME

Follow the steps in this procedure to register your system with an activation key. You can get the activation key from your organization administrator.

Prerequisites

• Activation key or keys.
See the Activation Keys page for creating new activation keys.

Procedure

1. Open the system menu, which is accessible from the upper-right screen corner, and click Settings.

2. Go to About → Subscription.

3. If you are not using the Red Hat server:
   a. In the Registration Server section, select Custom Address.
   b. Enter the server address in the URL field.

4. In the Registration Type section, select Activation Keys.

5. Under Registration Details:
   • Enter your activation keys in the Activation Keys field.
Separate your keys by a comma (,).

- Enter the name or ID of your organization in the **Organization** field.

6. Click **Register**.
CHAPTER 5. MAKING SYSTEMD SERVICES START AT BOOT TIME

As a system administrator, you can determine how services start on your system. For managing services, you can use the `systemctl` command-line utility for controlling the `systemd` system and service manager, or you can use the RHEL web console.

5.1. ENABLING OR DISABLING SERVICES

As a system administrator, you can enable or disable a service to start at boot, these changes apply with the next reboot. If you want a service to start automatically at boot, you must enable this service. If you disable a service, it will not start at boot, but you can start it manually. You can also mask a service, so that it cannot be started manually. Masking is a way of disabling a service that makes the service permanently unusable until it is unmasked again.

Prerequisites

- You must have root access to the system.
- The service you want to enable must not be masked. If you have a masked service, you must unmask it first:

```bash
# systemctl unmask service_name
```

Procedure

1. Enable a service to start at boot:

```bash
# systemctl enable service_name
```

Replace `service_name` with the service you want to enable.

You can also enable and start a service in a single command:

```bash
# systemctl enable --now service_name
```

2. Disable a service to start at boot:

```bash
# systemctl disable service_name
```

Replace `service_name` with the service you want to disable.

If you want to make a service permanently unusable, mask the service:

```bash
# systemctl mask service_name
```
CHAPTER 6. CONFIGURING SYSTEM SECURITY

Computer security is the protection of computer systems and their hardware, software, information, and services from theft, damage, disruption, and misdirection. Ensuring computer security is an essential task, in particular in enterprises that process sensitive data and handle business transactions.

This section covers only the basic security features that you can configure after installation of the operating system.

6.1. ENABLING THE FIREWALLD SERVICE

A firewall is a network security system that monitors and controls incoming and outgoing network traffic according to configured security rules. A firewall typically establishes a barrier between a trusted secure internal network and another outside network.

The `firewalld` service, which provides a firewall in Red Hat Enterprise Linux, is automatically enabled during installation.

To enable the `firewalld` service, follow this procedure.

**Procedure**

- Display the current status of `firewalld`:
  
  ```
  $ systemctl status firewalld
  ● firewalld.service - firewalld - dynamic firewall daemon
    Loaded: loaded (/usr/lib/systemd/system/firewalld.service; disabled; vendor preset: enabled)
    Active: inactive (dead)
  ...
  ```

- If `firewalld` is not enabled and running, switch to the `root` user, and start the `firewalld` service and enable to start it automatically after the system restarts:

  ```
  # systemctl enable --now firewalld
  ```

**Verification steps**

- Check that `firewalld` is running and enabled:

  ```
  $ systemctl status firewalld
  ● firewalld.service - firewalld - dynamic firewall daemon
    Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor preset: enabled)
    Active: active (running)
  ...
  ```

**Additional resources**

- Using and configuring `firewalld`

- `man firewalld(1)`
6.2. MANAGING BASIC SELINUX SETTINGS

Security-Enhanced Linux (SELinux) is an additional layer of system security that determines which processes can access which files, directories, and ports. These permissions are defined in SELinux policies. A policy is a set of rules that guide the SELinux security engine.

SELinux has two possible states:

- Disabled
- Enabled

When SELinux is enabled, it runs in one of the following modes:

- Enabled
  - Enforcing
  - Permissive

In **enforcing mode**, SELinux enforces the loaded policies. SELinux denies access based on SELinux policy rules and enables only the interactions that are explicitly allowed. Enforcing mode is the safest SELinux mode and is the default mode after installation.

In **permissive mode**, SELinux does not enforce the loaded policies. SELinux does not deny access, but reports actions that break the rules to the `/var/log/audit/audit.log` log. Permissive mode is the default mode during installation. Permissive mode is also useful in some specific cases, for example when troubleshooting problems.

**Additional resources**

- Using SELinux

6.3. ENSURING THE REQUIRED STATE OF SELINUX

By default, SELinux operates in enforcing mode. However, in specific scenarios, you can set SELinux to permissive mode or even disable it.

**IMPORTANT**

Red Hat recommends to keep your system in enforcing mode. For debugging purposes, you can set SELinux to permissive mode.

Follow this procedure to change the state and mode of SELinux on your system.

**Procedure**

1. Display the current SELinux mode:

   ```
   $ getenforce
   ```

2. To temporarily set SELinux:
   a. To Enforcing mode:
# setenforce Enforcing

b. To Permissive mode:

# setenforce Permissive

NOTE

After reboot, SELinux mode is set to the value specified in the /etc/selinux/config configuration file.

3. To set SELinux mode to persist across reboots, modify the SELINUX variable in the /etc/selinux/config configuration file.

   For example, to switch SELinux to enforcing mode:

   ```
   # This file controls the state of SELinux on the system.
   # SELINUX= can take one of these three values:
   #   enforcing - SELinux security policy is enforced.
   #   permissive - SELinux prints warnings instead of enforcing.
   #   disabled - No SELinux policy is loaded.
   SELINUX=enforcing
   ...
   ```

WARNING

Disabling SELinux reduces your system security. Avoid disabling SELinux using the SELINUX=disabled option in the /etc/selinux/config file because this can result in memory leaks and race conditions causing kernel panics. Instead, disable SELinux by adding the selinux=0 parameter to the kernel command line. For more information, see Changing SELinux modes at boot time.

Additional resources

- Changing SELinux states and modes

6.4. ADDITIONAL RESOURCES

- Generating SSH key pairs
- Setting an OpenSSH server for key-based authentication
- Security hardening
- Using SELinux
- Securing networks
CHAPTER 7. GETTING STARTED WITH MANAGING USER ACCOUNTS

Red Hat Enterprise Linux is a multi-user operating system, which enables multiple users on different computers to access a single system installed on one machine. Every user operates under its own account, and managing user accounts thus represents a core element of Red Hat Enterprise Linux system administration.

The following are the different types of user accounts:

- **Normal user accounts:**
  Normal accounts are created for users of a particular system. Such accounts can be added, removed, and modified during normal system administration.

- **System user accounts:**
  System user accounts represent a particular applications identifier on a system. Such accounts are generally added or manipulated only at software installation time, and they are not modified later.

  **WARNING**

  System accounts are presumed to be available locally on a system. If these accounts are configured and provided remotely, such as in the instance of an LDAP configuration, system breakage and service start failures can occur.

For system accounts, user IDs below 1000 are reserved. For normal accounts, you can use IDs starting at 1000. However, the recommended practice is to assign IDs starting at 5000. For assigning IDs, see the `/etc/login.defs` file.

- **Group:**
  A group is an entity which ties together multiple user accounts for a common purpose, such as granting access to particular files.

7.1. MANAGING ACCOUNTS AND GROUPS USING COMMAND LINE TOOLS

This section describes basic command-line tools to manage user accounts and groups.

- To display user and group IDs:

  $ id
  uid=1000(example.user) gid=1000(example.user) groups=1000(example.user),10(wheel)
  context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023

- To create a new user account:

  # useradd example.user
To assign a new password to a user account belonging to example.user:

```
# passwd example.user
```

To add a user to a group:

```
# usermod -a -G example.group example.user
```

Additional resources

- `man useradd(8)`, `man passwd(1)`, and `man usermod(8)`

### 7.2. SYSTEM USER ACCOUNTS MANAGED IN THE WEB CONSOLE

With user accounts displayed in the RHEL web console you can:

- Authenticate users when accessing the system.
- Set the access rights to the system.

The RHEL web console displays all user accounts located in the system. Therefore, you can see at least one user account just after the first login to the web console.

After logging into the RHEL web console, you can perform the following operations:

- Create new users accounts.
- Change their parameters.
- Lock accounts.
- Terminate user sessions.

### 7.3. ADDING NEW ACCOUNTS USING THE WEB CONSOLE

Use the following steps for adding user accounts to the system and setting administration rights to the accounts through the RHEL web console.

**Prerequisites**

- The RHEL web console must be installed and accessible. For details, see [Installing the web console](#).

**Procedure**

1. Log in to the RHEL web console.
2. Click **Accounts**.
3. Click **Create New Account**.
4. In the **Full Name** field, enter the full name of the user.
The RHEL web console automatically suggests a user name from the full name and fills it in the User Name field. If you do not want to use the original naming convention consisting of the first letter of the first name and the whole surname, update the suggestion.

5. In the Password/Confirm fields, enter the password and retype it for verification that your password is correct. The color bar below the fields shows you the security level of the entered password, which does not allow you to create a user with a weak password.

6. Click Create to save the settings and close the dialog box.

7. Select the newly created account.

8. Select Server Administrator in the Roles item.

Now you can see the new account in the Accounts settings and you can use its credentials to connect to the system.
CHAPTER 8. DUMPING A CRASHED KERNEL FOR LATER ANALYSIS

To analyze why a system crashed, you can use the **kdump** service to save the contents of the system’s memory for later analysis. This section provides a brief introduction to **kdump**, and information about configuring **kdump** using the RHEL web console or using the corresponding RHEL system role.

8.1. WHAT IS KDUMP

**kdump** is a service which provides a crash dumping mechanism. The service enables you to save the contents of the system memory for analysis. **kdump** uses the **kexec** system call to boot into the second kernel (a **capture kernel**) without rebooting; and then captures the contents of the crashed kernel’s memory (a **crash dump** or a **vmcore**) and saves it into a file. The second kernel resides in a reserved part of the system memory.

**IMPORTANT**

A kernel crash dump can be the only information available in the event of a system failure (a critical bug). Therefore, operational **kdump** is important in mission-critical environments. Red Hat advise that system administrators regularly update and test **kexec-tools** in your normal kernel update cycle. This is especially important when new kernel features are implemented.

You can enable **kdump** for all installed kernels on a machine or only for specified kernels. This is useful when there are multiple kernels used on a machine, some of which are stable enough that there is no concern that they could crash.

When **kdump** is installed, a default `/etc/kdump.conf` file is created. The file includes the default minimum **kdump** configuration. You can edit this file to customize the **kdump** configuration, but it is not required.

8.2. CONFIGURING KDUMP MEMORY USAGE AND TARGET LOCATION IN WEB CONSOLE

The procedure below shows you how to use the **Kernel Dump** tab in the RHEL web console interface to configure the amount of memory that is reserved for the **kdump** kernel. The procedure also describes how to specify the target location of the **vmcore** dump file and how to test your configuration.

**Procedure**

1. Open the **Kernel Dump** tab and start the **kdump** service.

2. Configure the **kdump** memory usage using the command line.

3. Click the link next to the **Crash dump location** option.
4. Select the **Local Filesystem** option from the drop-down and specify the directory you want to save the dump in.

- Alternatively, select the **Remote over SSH** option from the drop-down to send the vmcore to a remote machine using the SSH protocol. Fill the **Server**, **ssh key**, and **Directory** fields with the remote machine address, ssh key location, and a target directory.

- Another choice is to select the **Remote over NFS** option from the drop-down and fill the **Mount** field to send the vmcore to a remote machine using the NFS protocol.

**NOTE**

Tick the **Compression** check box to reduce the size of the vmcore file.

5. Test your configuration by crashing the kernel.
a. Click **Test configuration**.

b. In the **Test kdump settings** field, click **Crash system**.

**WARNING**

This step disrupts execution of the kernel and results in a system crash and loss of data.

**Additional resources**

- Supported kdump targets
- Using secure communications between two systems with OpenSSH

## 8.3. KDUMP USING RHEL SYSTEM ROLES

RHEL System Roles is a collection of Ansible roles and modules that provide a consistent configuration interface to remotely manage multiple RHEL systems. The **kdump** role enables you to set basic kernel dump parameters on multiple systems.

**WARNING**

The **kdump** role replaces the **kdump** configuration of the managed hosts entirely by replacing the /etc/kdump.conf file. Additionally, if the **kdump** role is applied, all previous **kdump** settings are also replaced, even if they are not specified by the role variables, by replacing the /etc/sysconfig/kdump file.
The following example playbook shows how to apply the *kdump* system role to set the location of the crash dump files:

```yaml
---
- hosts: kdump-test
  vars:
    kdump_path: /var/crash
  roles:
    - rhel-system-roles.kdump
```

For a detailed reference on *kdump* role variables, install the *rhel-system-roles* package, and see the README.md or README.html files in the `/usr/share/doc/rhel-system-roles/kdump` directory.

## Additional resources

- [Introduction to RHEL System Roles](#)

### 8.4. ADDITIONAL RESOURCES

- [Installing kdump](#)
- [Configuring kdump on the command line](#)
- [Configuring kdump in the web console](#)
CHAPTER 9. RECOVERING AND RESTORING A SYSTEM

To recover and restore a system using an existing backup, Red Hat Enterprise Linux provides the Relax-and-Recover (ReaR) utility.

You can use the utility as a disaster recovery solution and also for system migration.

The utility enables you to perform the following tasks:

- Produce a bootable image and restore the system from an existing backup, using the image.
- Replicate the original storage layout.
- Restore user and system files.
- Restore the system to a different hardware.

Additionally, for disaster recovery, you can also integrate certain backup software with ReaR.

Setting up ReaR involves the following high-level steps:

1. Install ReaR.
3. Create rescue system.
4. Generate backup files.

9.1. SETTING UP REAR

Use the following steps to install the package for using the Relax-and-Recover (ReaR) utility, create a rescue system, configure and generate a backup.

Prerequisites

- Necessary configurations as per the backup restore plan are ready.
  Note that you can use the `NETFS` backup method, a fully-integrated and built-in method with ReaR.

Procedure

1. Install the ReaR utility by running the following command:

   ```
   # dnf install rear
   ```

2. Modify the ReaR configuration file in an editor of your choice, for example:

   ```
   # vi /etc/rear/local.conf
   ```

3. Add the backup setting details to `/etc/rear/local.conf`. For example, in the case of the `NETFS` backup method, add the following lines:

   ```
   BACKUP=NETFS
   BACKUP_URL=backup.location
   ```
Replace `backup.location` by the URL of your backup location.

4. To configure ReaR to keep the previous backup archive when the new one is created, also add the following line to the configuration file:

   ```
   NETFS_KEEP_OLD_BACKUP_COPY=y
   ```

5. To make the backups incremental, meaning that only the changed files are backed up on each run, add the following line:

   ```
   BACKUP_TYPE=incremental
   ```

6. Create a rescue system:

   ```
   # rear mkrescue
   ```

7. Take a backup as per the restore plan. For example, in the case of the `NETFS` backup method, run the following command:

   ```
   # rear mkbackuponly
   ```

   Alternatively, you can create the rescue system and the backup in a single step by running the following command:

   ```
   # rear mkbackup
   ```

   This command combines the functionality of the `rear mkrescue` and `rear mkbackuponly` commands.

### 9.2. USING A REAR RESCUE IMAGE ON THE 64-BIT IBM Z ARCHITECTURE

Basic Relax and Recover (ReaR) functionality is now available on the 64-bit IBM Z architecture as a Technology Preview. You can create a ReaR rescue image on IBM Z only in the z/VM environment. Backing up and recovering logical partitions (LPARs) has not been tested.

**IMPORTANT**

ReaR on the 64-bit IBM Z architecture is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process. For more information about the support scope of Red Hat Technology Preview features, see [https://access.redhat.com/support/offerings/techpreview](https://access.redhat.com/support/offerings/techpreview).

The only output method currently available is Initial Program Load (IPL). IPL produces a kernel and an initial ramdisk (initrd) that can be used with the `ziPL` bootloader.

**Prerequisites**
- ReaR is installed.
  - To install ReaR, run the `dnf install rear` command

**Procedure**

Add the following variables to the `/etc/rear/local.conf` to configure ReaR for producing a rescue image on the 64-bit IBM Z architecture:

1. To configure the IPL output method, add `OUTPUT=IPL`.

2. To configure the backup method and destination, add `BACKUP` and `BACKUP_URL` variables. For example:
   
   ```
   BACKUP=NETFS
   BACKUP_URL=nfs://<nfsserver name>/<share path>
   ```

   **IMPORTANT**
   
The local backup storage is currently not supported on the 64-bit IBM Z architecture.

3. Optionally, you can also configure the `OUTPUT_URL` variable to save the kernel and `initrd` files. By default, the `OUTPUT_URL` is aligned with `BACKUP_URL`.

4. To perform backup and rescue image creation:

   ```
   rear mkbackup
   ```

5. This creates the kernel and initrd files at the location specified by the `BACKUP_URL` or `OUTPUT_URL` (if set) variable, and a backup using the specified backup method.

6. To recover the system, use the ReaR kernel and initrd files created in step 3, and boot from a Direct Attached Storage Device (DASD) or a Fibre Channel Protocol (FCP)-attached SCSI device prepared with the `zipl` boot loader, kernel, and `initrd`. For more information, see Using a Prepared DASD.

7. When the rescue kernel and `initrd` get booted, it starts the ReaR rescue environment. Proceed with system recovery.

---

**WARNING**

Currently, the rescue process reformats all the DASDs (Direct Attached Storage Devices) connected to the system. Do not attempt a system recovery if there is any valuable data present on the system storage devices. This also includes the device prepared with the `zipl` bootloader, ReaR kernel, and `initrd` that were used to boot into the rescue environment. Ensure to keep a copy.

---

**Additional resources**
• Installing under z/VM
• Using a Prepared DASD
CHAPTER 10. TROUBLESHOOTING PROBLEMS USING LOG FILES

Log files contain messages about the system, including the kernel, services, and applications running on it. These contain information that helps troubleshoot issues or monitor system functions. The logging system in Red Hat Enterprise Linux is based on the built-in syslog protocol. Particular programs use this system to record events and organize them into log files, which are useful when auditing the operating system and troubleshooting various problems.

10.1. SERVICES HANDLING SYSLOG MESSAGES

The following two services handle syslog messages:

- The systemd-journald daemon
- The Rsyslog service

The systemd-journald daemon collects messages from various sources and forwards them to Rsyslog for further processing. The systemd-journald daemon collects messages from the following sources:

- Kernel
- Early stages of the boot process
- Standard and error output of daemons as they start up and run
- Syslog

The Rsyslog service sorts the syslog messages by type and priority and writes them to the files in the /var/log directory. The /var/log directory persistently stores the log messages.

10.2. SUBDIRECTORIES STORING SYSLOG MESSAGES

The following subdirectories under the /var/log directory store syslog messages.

- /var/log/messages - all syslog messages except the following
- /var/log/secure - security and authentication-related messages and errors
- /var/log/maillog - mail server-related messages and errors
- /var/log/cron - log files related to periodically executed tasks
- /var/log/boot.log - log files related to system startup

10.3. INSPECTING LOG FILES USING THE WEB CONSOLE

Follow the steps in this procedure to inspect the log files using the RHEL web console.

Procedure

1. Log into the RHEL web console. For details see Logging in to the web console.
2. Click Logs.
### 10.4. VIEWING LOGS USING THE COMMAND LINE

The Journal is a component of systemd that helps to view and manage log files. It addresses problems connected with traditional logging, closely integrated with the rest of the system, and supports various logging technologies and access management for the log files.

You can use the `journalctl` command to view messages in the system journal using the command line, for example:

```bash
$ journalctl -b | grep kvm
May 15 11:31:41 localhost.localdomain kernel: kvm-clock: Using msrs 4b564d01 and 4b564d00
May 15 11:31:41 localhost.localdomain kernel: kvm-clock: cpu 0, msr 76401001, primary cpu clock...
```

### Table 10.1. Viewing system information

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>journalctl</code></td>
<td>Shows all collected journal entries.</td>
</tr>
<tr>
<td><code>journalctl FILEPATH</code></td>
<td>Shows logs related to a specific file. For example, the <code>journalctl /dev/sda</code> command displays logs related to the <code>/dev/sda</code> file system.</td>
</tr>
<tr>
<td><code>journalctl -b</code></td>
<td>Shows logs for the current boot.</td>
</tr>
<tr>
<td><code>journalctl -k -b -1</code></td>
<td>Shows kernel logs for the current boot.</td>
</tr>
</tbody>
</table>

### Table 10.2. Viewing information on specific services

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>journalctl -b _SYSTEMD_UNIT=foo</code></td>
<td>Filters log to see ones matching the &quot;foo&quot; systemd service.</td>
</tr>
<tr>
<td><code>journalctl -b _SYSTEMD_UNIT=foo _PID=number</code></td>
<td>Combines matches. For example, this command shows logs for systemd-units that match foo and the PID number.</td>
</tr>
</tbody>
</table>
The separator “+” combines two expressions in a logical OR. For example, this command shows all messages from the `foo` service process with the PID plus all messages from the `foo1` service (from any of its processes).

This command shows all entries matching either expression, referring to the same field. Here, this command shows logs matching a systemd-unit `foo` or a systemd-unit `foo1`.

Table 10.3. Viewing logs related to specific boots

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>journalctl --list-boots</code></td>
<td>Shows a tabular list of boot numbers, their IDs, and the timestamps of the first and last message pertaining to the boot. You can use the ID in the next command to view detailed information.</td>
</tr>
<tr>
<td><code>journalctl --boot=ID _SYSTEMD_UNIT=foo</code></td>
<td>Shows information about the specified boot ID.</td>
</tr>
</tbody>
</table>

10.5. ADDITIONAL RESOURCES

- man `journalctl(1)`
- Configuring a remote logging solution
CHAPTER 11. ACCESSING THE RED HAT SUPPORT

This section describes how to effectively troubleshoot your problems using Red Hat support and sosreport.

To obtain support from Red Hat, use the Red Hat Customer Portal, which provides access to everything available with your subscription.

11.1. OBTAINING RED HAT SUPPORT THROUGH RED HAT CUSTOMER PORTAL

The following section describes how to use the Red Hat Customer Portal to get help.

Prerequisites

- A valid user account on the Red Hat Customer Portal. See Create a Red Hat Login.
- An active subscription for the RHEL system.

Procedure

1. Access Red Hat support:
   a. Open a new support case.
   b. Initiate a live chat with a Red Hat expert.
   c. Contact a Red Hat expert by making a call or sending an email.

11.2. TROUBLESHOOTING PROBLEMS USING SOSREPORT

The sosreport command collects configuration details, system information and diagnostic information from a Red Hat Enterprise Linux system.

The following section describes how to use the sosreport command to produce reports for your support cases.

Prerequisites

- A valid user account on the Red Hat Customer Portal. See Create a Red Hat Login.
- An active subscription for the RHEL system.
- A support-case number.

Procedure

1. Install the sos package:

   ```
   # dnf install sos
   ```
NOTE

The default minimal installation of Red Hat Enterprise Linux does not include the `sos` package, which provides the `sosreport` command.

2. Generate a report:

   ```bash
   # sosreport
   ```

3. Attach the report to your support case.
   See the [How can I attach a file to a Red Hat support case?](https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/9/html/Configuring_basic_system_settings/) Red Hat Knowledgebase article for more information.

   Note that when attaching the report, you are prompted to enter the number of the relevant support case.

Additional resources

- [What is an sosreport and how to create one in Red Hat Enterprise Linux?](https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/9/html/Configuring_basic_system_settings/)
CHAPTER 12. INTRODUCTION TO SYSTEMD

As a system administrator, you need to interact with `systemd`, the system and service manager for Linux operating systems. The `systemd` software suite provides tools and services to control and report the state of the system, to initialize your system during start and many more. Since Red Hat Enterprise Linux 7, `systemd` is a replacement for Upstart as the default init system, and is backwards compatible with SysV init scripts. The `systemd` software suite provides a number of features such as:

- parallel start of system services at boot time,
- on-demand activation of daemons,
- dependency-based service control logic.

As a representation of system resources and services, `systemd` introduces the concept of `systemd units`. A systemd unit, which performs or controls a particular task, is the basic object that systemd manages. See the following examples of various systemd unit types:

- service,
- target,
- device,
- mount,
- timer,
- other types that are relevant to the init system.

**NOTE**

If you want to display all available unit types, use:

```
# systemctl -t help
```

A systemd unit consists of a name, type and a configuration file, which defines the task of the unit. The unit configuration files are located in one of the directories listed in the following table:

**Table 12.1. systemd unit files locations**

<table>
<thead>
<tr>
<th>Directory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>/usr/lib/systemd/system/</code></td>
<td>systemd unit files distributed with installed RPM packages.</td>
</tr>
<tr>
<td><code>/run/systemd/system/</code></td>
<td>systemd unit files created at run time. This directory takes precedence over the directory with installed service unit files.</td>
</tr>
</tbody>
</table>
The default configuration of `systemd` is defined during the compilation and you can find the configuration in the `/etc/systemd/system.conf` file. Use this file if you want to deviate from those defaults and override selected default values for systemd units globally.

For example, to override the default value of the timeout limit, which is set to 90 seconds, use the `DefaultTimeoutStartSec` parameter to input the required value in seconds.

```
DefaultTimeoutStartSec=required value
```
CHAPTER 13. MANAGING SYSTEM SERVICES WITH SYSTEMCTL

As a system administrator, you want to manage system services and perform different tasks related to different services, such as starting, stopping, restarting, enabling, and disabling services, listing services, and displaying system services statuses. To interact with the systemctl utility.

13.1. LISTING SYSTEM SERVICES

You can list all currently loaded service units and the status of all available service units.

Procedure

- To list all currently loaded service units, enter:

```
$ systemctl list-units --type service
```

<table>
<thead>
<tr>
<th>UNIT</th>
<th>LOAD</th>
<th>ACTIVE</th>
<th>SUB</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>abrt-ccpp.service</td>
<td>loaded</td>
<td>active</td>
<td>exited</td>
<td>Install ABRT coredump hook</td>
</tr>
<tr>
<td>abrt-oops.service</td>
<td>loaded</td>
<td>active</td>
<td>running</td>
<td>ABRT kernel log watcher</td>
</tr>
<tr>
<td>abrd.service</td>
<td>loaded</td>
<td>active</td>
<td>running</td>
<td>ABRT Automated Bug Reporting Tool</td>
</tr>
<tr>
<td>systemd-vconsole-setup.service</td>
<td>loaded</td>
<td>active</td>
<td>exited</td>
<td>Setup Virtual Console</td>
</tr>
<tr>
<td>tog-pegasus.service</td>
<td>loaded</td>
<td>active</td>
<td>running</td>
<td>OpenPegasus CIM Server</td>
</tr>
</tbody>
</table>

- To list all loaded units regardless of their state, enter the following command with the --all or -a command line option:

```
$ systemctl list-units --type service --all
```

- To list the status (enabled or disabled) of all available service units, enter:

```
$ systemctl list-unit-files --type service
```

<table>
<thead>
<tr>
<th>UNIT FILE</th>
<th>STATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>abrt-ccpp.service</td>
<td>enabled</td>
</tr>
<tr>
<td>abrt-oops.service</td>
<td>enabled</td>
</tr>
</tbody>
</table>
For each service unit, this command displays:

- **UNIT FILE**: its full name
- **STATE**: information whether the service unit is enabled or disabled

### Additional resources

- [Displaying system service status](#)

## 13.2. DISPLAYING SYSTEM SERVICE STATUS

You can inspect any service unit to get its detailed information and verify the state of the service, whether it is enabled or running. You can also view services that are ordered to start after or before a particular service unit.

### Procedure

- To display detailed information about a service unit that corresponds to a system service, enter:

  ```shell
  $ systemctl status <name>.service
  ```

  Replace `<name>` with the name of the service unit you want to inspect (for example, `gdm`).

  This command displays the name of the selected service unit followed by its short description, one or more fields described in [Available service unit information](#), if it is executed by the **root** user, and the most recent log entries.

### Table 13.1. Available service unit information

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Loaded</strong></td>
<td>Information whether the service unit has been loaded, the absolute path to</td>
</tr>
<tr>
<td></td>
<td>the unit file, and a note whether the unit is enabled.</td>
</tr>
<tr>
<td><strong>Active</strong></td>
<td>Information whether the service unit is running followed by a time stamp.</td>
</tr>
<tr>
<td><strong>Main PID</strong></td>
<td>The PID of the corresponding system service followed by its name.</td>
</tr>
<tr>
<td><strong>Status</strong></td>
<td>Additional information about the corresponding system service.</td>
</tr>
</tbody>
</table>
Example 13.1. Displaying service status

The service unit for the GNOME Display Manager is named `gdm.service`. To determine the current status of this service unit, type the following at a shell prompt:

```
# systemctl status gdm.service
```

```
gdm.service - GNOME Display Manager
   Loaded: loaded (/usr/lib/systemd/system/gdm.service; enabled)
   Active: active (running) since Thu 2013-10-17 17:31:23 CEST; 5min ago
   Main PID: 1029 (gdm)
   CGroup: /system.slice/gdm.service
          └─1029 /usr/sbin/gdm
```

Oct 17 17:31:23 localhost systemd[1]: Started GNOME Display Manager.

- To only verify that a particular service unit is running, enter:
  
  `$ systemctl is-active <name>.service`

- To determine whether a particular service unit is enabled, enter:
  
  `$ systemctl is-enabled <name>.service`

**NOTE**  
Both `systemctl is-active` and `systemctl is-enabled` return an exit status of **0** if the specified service unit is running or enabled.

- To determine what services are ordered to start before the specified service unit, enter:
  
  `# systemctl list-dependencies --after <name>.service`

Replace `<name>` with the name of the service in the command.

For example, to view the list of services ordered to start before `gdm`, enter:

```
# systemctl list-dependencies --after gdm.service
```

```
gdm.service
   └─dbus.socket
   └─getty@tty1.service
```

To determine what services are ordered to start after the specified service unit, enter:

```
# systemctl list-dependencies --before <name>.service
```

Replace `<name>` with the name of the service in the command.

For example, to view the list of services ordered to start after `gdm`, enter:

```
# systemctl list-dependencies --before gdm.service
```

```
gdm.service
└─ dracut-shutdown.service
    └─ shutdown.target
      └─ systemd-reboot.service
```

Additional resources

- Listing system services

**13.3. STARTING A SYSTEM SERVICE**

You can start system service in the current session using the **start** command.

**Prerequisites**

- You must have root access to the system.

**Procedure**

- To start a selected service unit corresponding to a system service, type the following command as **root**:

  ```
  # systemctl start <name>.service
  ```

Replace `<name>` with the name of the service unit you want to start (for example, `httpd.service`).

**Example 13.2. Starting httpd.service**

---

60
The service unit for the Apache HTTP Server is named `httpd.service`. To activate this service unit and start the `httpd` daemon in the current session, enter the following command as `root`:

```
# systemctl start httpd.service
```

**NOTE**

In `systemd`, positive and negative dependencies between services exist. Starting a particular service may require starting one or more other services (positive dependency) or stopping one or more services (negative dependency).

When you attempt to start a new service, `systemd` resolves all dependencies automatically, without explicit notification to the user. This means that if you are already running a service, and you attempt to start another service with a negative dependency, the first service is automatically stopped.

For example, if you are running the `postfix` service, and you attempt to start the `sendmail` service, `systemd` first automatically stops `postfix`, because these two services are conflicting and cannot run on the same port.

Additional resources

- Positive and negative service dependencies
- Enabling a system service
- Displaying system service status

### 13.4. STOPPING A SYSTEM SERVICE

If you want to stop a system service in the current session, use the `stop` command.

**Prerequisites**

- You must have root access to the system.

**Procedure**

- To stop the service unit corresponding to a system service, enter the following command as `root`:

```
# systemctl stop <name>.service
```

Replace `<name>` with the name of the service unit you want to stop (for example, `bluetooth`).

**Example 13.3. Stopping bluetoothd.service**

The service unit for the `bluetoothd` daemon is named `bluetooth.service`. To deactivate this service unit and stop the `bluetoothd` daemon in the current session, enter the following command as `root`:

```
# systemctl stop bluetooth.service
```
13.5. RESTARTING A SYSTEM SERVICE

You can restart system service in the current session using the `restart` command.

This procedure describes how to:

- Stop the selected service unit in the current session and immediately start it again
- Restart a service unit only if the corresponding service is already running
- Reload configuration of a system service without interrupting its execution

**Prerequisites**

- You must have root access to the system.

**Procedure**

- Restart a service unit corresponding to a system service:

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

  Replace `<name>` with the name of the service unit you want to restart (for example, `httpd`).

  **NOTE**

  If the selected service unit is not running, this command starts it too.

- Alternatively, restart a service unit only if the corresponding service is already running:

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

- Alternatively, reload the configuration without interrupting service execution:

  ```
  # systemctl reload <name>.service
  ```

  **NOTE**

  System services that do not support this feature, ignore this command. To restart such services, use the `reload-or-restart` and `reload-or-try-restart` commands instead.

**Example 13.4. Reloading httpd.service**

In order to prevent users from encountering unnecessary error messages or partially
rendered web pages, the Apache HTTP Server allows you to edit and reload its configuration without the need to restart it and interrupt actively processed requests. To do so, use the following command:

```
# systemctl reload httpd.service
```

### Additional resources

- [Displaying system service status](#)

### 13.6. ENABLING A SYSTEM SERVICE

You can configure service to start automatically at the system booting time. The `enable` command reads the `[Install]` section of the selected service unit and creates appropriate symbolic links to the `/usr/lib/systemd/system/` name.service file in the `/etc/systemd/system/` directory and its subdirectories. However, it does not rewrite links that already exist.

#### Prerequisites

- You must have root access to the system.

#### Procedure

- Configure a service unit that corresponds to a system service to be automatically started at boot time:

  ```
  # systemctl enable <name>.service
  
  Replace `<name>` with the name of the service unit you want to enable (for example, `httpd`).
  ```

- Alternatively, if you want to ensure that the symbolic links are re-created, reenable the system unit:

  ```
  # systemctl reenable <name>.service
  
  This command disables the selected service unit and immediately enables it again.
  ```

**Example 13.5. Enabling httpd.service**

To configure the Apache HTTP Server to start automatically at boot time, use the following command:

```
# systemctl enable httpd.service
Created symlink from /etc/systemd/system/multi-user.target.wants/httpd.service to /usr/lib/systemd/system/httpd.service.
```

### Additional resources

- [Displaying system service status](#)
13.7. DISABLING A SYSTEM SERVICE

You can prevent a service unit from starting automatically at boot time. The `disable` command reads the [Install] section of the selected service unit and removes appropriate symbolic links to the `/usr/lib/systemd/system/name.service` file from the `/etc/systemd/system/` directory and its subdirectories.

Prerequisites

- You must have root access to the system.

Procedure

- To configure a service unit that corresponds to a system service not to start automatically at boot time, enter the following command as `root`:

  ```
  # systemctl disable <name>.service
  ```

  Replace `<name>` with the name of the service unit you want to disable (for example, `bluetooth`).

  **Example 13.6. Disabling bluetoothd.service**

  The service unit for the `bluetoothd` daemon is named `bluetooth.service`. To prevent this service unit from starting at boot time, enter the following command as a `root`:

  ```
  # systemctl disable bluetooth.service
  ```

  Removed symlink `/etc/systemd/system/bluetooth.target.wants/bluetooth.service`.
  Removed symlink `/etc/systemd/system/dbus-org.bluez.service`.

  Alternatively, you can mask any service unit and prevent it from being started manually or by another service:

  ```
  # systemctl mask <name>.service
  ```

  This command replaces the `/etc/systemd/system/name.service` file with a symbolic link to `/dev/null`, rendering the actual unit file inaccessible to `systemd`.

  - To revert this action and unmask a service unit, enter:

    ```
    # systemctl unmask <name>.service
    ```

Additional resources

- Displaying system service status
- Stopping a system service
CHAPTER 14. WORKING WITH SYSTEMD TARGETS

Targets in **systemd** act as synchronization points during the start of your system. Target unit files, which end with the `.target` file extension, represent the **systemd** targets. The purpose of target units is to group together various **systemd** units through a chain of dependencies.

Consider the following examples:

- The **graphical.target** unit for starting a graphical session, starts system services such as the GNOME Display Manager (**gdm.service**) or Accounts Service (**accounts-daemon.service**), and also activates the **multi-user.target** unit.

- Similarly, the **multi-user.target** unit starts other essential system services such as NetworkManager (**NetworkManager.service**) or D-Bus (**dbus.service**) and activates another target unit named **basic.target**.

While working with **systemd** targets, you can view the default target, change it or change the current target.

### 14.1. VIEWING THE DEFAULT TARGET

You can display the default target using the `systemctl` command or examine the `/etc/systemd/system/default.target` file, which represents the default target unit.

**Procedure**

- Determine, which target unit is used by default:
  
  ```
  $ systemctl get-default
  graphical.target
  ```

- Determine the default target using the symbolic link:
  
  ```
  $ ls -l /usr/lib/systemd/system/default.target
  ```

### 14.2. VIEWING THE TARGET UNITS

You can display all unit types or limit your search to the currently loaded target units. By default, the `systemctl list-units` command displays only active units.

**Procedure**

- List all loaded units regardless of their state:
  
  ```
  $ systemctl list-units --type target --all
  ```

- Alternatively, list all currently loaded target units:
  
  ```
  $ systemctl list-units --type target
  ```

<table>
<thead>
<tr>
<th>UNIT</th>
<th>LOAD</th>
<th>ACTIVE</th>
<th>SUB</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic.target</td>
<td>loaded</td>
<td>active</td>
<td>active</td>
<td>Basic System</td>
</tr>
<tr>
<td>cryptsetup.target</td>
<td>loaded</td>
<td>active</td>
<td>active</td>
<td>Encrypted Volumes</td>
</tr>
</tbody>
</table>
14.3. CHANGING THE DEFAULT TARGET

The default target unit is represented by the `/etc/systemd/system/default.target` file. The following procedure describes how to change the default target by using the `systemctl` command:

Procedure

1. To determine the default target unit:

   ```
   # systemctl get-default
   ```

2. To configure the system to use a different target unit by default:

   ```
   # systemctl set-default multi-user.target
   rm /etc/systemd/system/default.target
   ln -s /usr/lib/systemd/system/multi-user.target /etc/systemd/system/default.target
   ```

This command replaces the `/etc/systemd/system/default.target` file with a symbolic link to `/usr/lib/systemd/system/name.target`, where `name` is the name of the target unit you want to use. Replace `multi-user` with the name of the target unit you want to use by default.

Table 14.1. Common targets for `set-default` command

<table>
<thead>
<tr>
<th>basic</th>
<th>unit target covering basic boot-up</th>
</tr>
</thead>
<tbody>
<tr>
<td>rescue</td>
<td>unit target that pulls in the base system and spawns a rescue shell</td>
</tr>
<tr>
<td>multi-user</td>
<td>unit target for setting up a multi-user system</td>
</tr>
<tr>
<td>graphical</td>
<td>unit target for setting up a graphical login screen</td>
</tr>
<tr>
<td>emergency</td>
<td>unit target that starts an emergency shell on the main console</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------------------------------------------------------</td>
</tr>
<tr>
<td>sysinit</td>
<td>unit target that pulls in the services required for system initialization</td>
</tr>
</tbody>
</table>

3. Reboot

```bash
# reboot
```

Additional resources

- `systemd.special` man page
- `bootup` man page

### 14.4. CHANGING THE DEFAULT TARGET USING A SYMBOLIC LINK

You can change the default target by creating a symbolic link to the target you want.

**Procedure**

1. Determine the default target unit:

   ```bash
   # ls -l /etc/systemd/system/default.target
   ```

   Note that in certain cases, the `/etc/systemd/system/default.target` link might not exist, and systemd looks for the default target unit in `/usr`. In such cases, determine the default target unit using the following command:

   ```bash
   # ls -l /usr/lib/systemd/system/default.target
   ```

2. Delete the `/etc/systemd/system/default.target` link:

   ```bash
   # rm /etc/systemd/system/default.target
   ```

3. Create a symbolic link:

   ```bash
   # ln -sf /usr/lib/systemd/system/graphical.target /etc/systemd/system/default.target
   ```

4. Reboot the system:

   ```bash
   # reboot
   ```

**Verification steps**

- Verify the newly created `default.target`:
14.5. CHANGING THE CURRENT TARGET

When you set a default target unit, the current target remains unchanged until the next reboot. If you want to change the target unit in the current session without reboot, use the `systemctl isolate` command.

Procedure

- Change to a different target unit in the current session:

  ```
  # systemctl isolate multi-user.target
  ```

  This command starts the target unit named `multi-user` and all dependent units, and immediately stops all others.

Replace `multi-user` with the name of the target unit you want to use by default.

Verification steps

- Verify the newly created default.target:

  ```
  $ systemctl get-default
  ```

14.6. BOOTING TO RESCUE MODE

`Rescue mode` provides a convenient single-user environment and allows you to repair your system in situations when it is unable to complete a regular booting process. In rescue mode, the system attempts to mount all local file systems and start certain important system services, but it does not activate network interfaces or allow more users to be logged into the system at the same time.

Procedure

- To enter the rescue mode, change the current target in the current session:

  ```
  # systemctl rescue
  ```

  Broadcast message from root@localhost on pts/0 (Fri 2013-10-25 18:23:15 CEST):

  The system is going down to rescue mode NOW!
NOTE

This command is similar to `systemctl isolate rescue.target`, but it also sends an informative message to all users that are currently logged into the system.

To prevent `systemd` from sending a message, run the following command with the `--no-wall` command-line option:

```
# systemctl --no-wall rescue
```

14.7. BOOTING TO EMERGENCY MODE

Emergency mode provides the most minimal environment possible and allows you to repair your system even in situations when the system is unable to enter rescue mode. In emergency mode, the system mounts the root file system only for reading, does not attempt to mount any other local file systems, does not activate network interfaces, and only starts a few essential services.

Procedure

- To enter the emergency mode, change the current target:

```
# systemctl emergency
```

NOTE

This command is similar to `systemctl isolate emergency.target`, but it also sends an informative message to all users that are currently logged into the system.

To prevent systemd from sending this message, run the following command with the `--no-wall` command-line option:

```
# systemctl --no-wall emergency
```
CHAPTER 15. SHUTTING DOWN, SUSPENDING, AND HIBERNATING THE SYSTEM

This section contains instructions about shutting down, suspending, or hibernating your operating system.

15.1. SYSTEM SHUTDOWN

To shut down the system, you can either use the systemctl utility directly, or call this utility through the shutdown command.

The advantage of using the shutdown command is:

- The support for time argument
  This is particularly useful for scheduled maintenance. Also, users have more time to react to the warning that a system shutdown has been scheduled.

- The option to cancel the shutdown

Additional resources

- Shutting down the system using the shutdown command
- Shutting down the system using the systemctl command
- Overview of the power management commands with systemctl

15.2. SHUTTING DOWN THE SYSTEM USING THE SHUTDOWN COMMAND

By following this procedure, you can use the shutdown command to perform various operations. You can either shut down the system and power off the machine at a certain time, or shut down and halt the system without powering off the machine, or cancel a pending shutdown.

Prerequisites

- Switch to the root user

Procedure

- To shut down the system and power off the machine at a certain time, use the command in the following format:

  ```
  shutdown --poweroff hh:mm
  ```

  Where \textit{hh:mm} is the time in 24 hour clock format. The \texttt{/run/nologin} file is created 5 minutes before system shutdown to prevent new logins.

  When a time argument is used, an optional \textit{wall message} can be appended to the command.

  Alternatively, to shut down and halt the system after a delay, without powering off the machine, use:
shutdown --halt +m

Where +m is the delay time in minutes. The now keyword is an alias for +0.

To cancel a pending shutdown, use:

shutdown -c

Additional resources

- shutdown(8) manual page
- Shutting down the system using the systemctl command

15.3. SHUTTING DOWN THE SYSTEM USING THE SYSTEMCTL COMMAND

By following this procedure, you can use the systemctl command to perform various operations. You can either shut down the system and power off the machine, or shut down and halt the system without powering off the machine.

Prerequisites

- Switch to the root user

Procedure

- To shut down the system and power off the machine, use the command in the following format:

  systemctl poweroff

  Alternatively, to shut down and halt the system without powering off the machine, use:

  systemctl halt

  NOTE

  By default, running either of these commands causes systemd to send an informative message to all users that are currently logged into the system. To prevent systemd from sending this message, run the selected command with the --no-wall command line option.

Additional resources

- Shutting down the system using the shutdown command

15.4. RESTARTING THE SYSTEM

You can restart the system by following this procedure.

Prerequisites
15.5. SUSPENDING THE SYSTEM

You can suspend the system by following this procedure.

Prerequisites

- Switch to the root user.

Procedure

- To suspend the system, run the following command:

  ```bash
  systemctl suspend
  ```

  This command saves the system state in RAM and with the exception of the RAM module, powers off most of the devices in the machine. When you turn the machine back on, the system then restores its state from RAM without having to boot again.

  Because the system state is saved in RAM and not on the hard disk, restoring the system from suspend mode is significantly faster than from hibernation. However, note that the suspended system state is also vulnerable to power outages.

Additional resources

- [Hibernating the system](#)

15.6. HIBERNATING THE SYSTEM

By following this procedure, you can either hibernate the system, or hibernate and suspend the system.

Prerequisites

- Switch to the root user.

Procedure

- To hibernate the system, run the following command:
systemctl hibernate

This command saves the system state on the hard disk drive and powers off the machine. When you turn the machine back on, the system then restores its state from the saved data without having to boot again.

Because the system state is saved on the hard disk and not in RAM, the machine does not have to maintain electrical power to the RAM module. However, as a consequence, restoring the system from hibernation is significantly slower than restoring it from suspend mode.

Alternatively, to hibernate and suspend the system, run the following command:

systemctl hybrid-sleep

Additional resources
- Suspending the system

15.7. OVERVIEW OF THE POWER MANAGEMENT COMMANDS WITH SYSTEMCTL

You can use the following list of the systemctl commands to control the power management of your system.

Table 15.1. Overview of the systemctl power management commands

<table>
<thead>
<tr>
<th>systemctl command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>systemctl halt</td>
<td>Halts the system.</td>
</tr>
<tr>
<td>systemctl poweroff</td>
<td>Powers off the system.</td>
</tr>
<tr>
<td>systemctl reboot</td>
<td>Restarts the system.</td>
</tr>
<tr>
<td>systemctl suspend</td>
<td>Suspends the system.</td>
</tr>
<tr>
<td>systemctl hibernate</td>
<td>Hibernates the system.</td>
</tr>
<tr>
<td>systemctl hybrid-sleep</td>
<td>Hibernates and suspends the system.</td>
</tr>
</tbody>
</table>

CHAPTER 15. SHUTTING DOWN, SUSPENDING, AND HIBERNATING THE SYSTEM
CHAPTER 16. WORKING WITH SYSTEMD UNIT FILES

This chapter includes the description of systemd unit files. The following sections show you how to:

- Create custom unit files
- Convert SysV init scripts to unit files
- Modify existing unit files
- Work with instantiated units

16.1. INTRODUCTION TO UNIT FILES

A unit file contains configuration directives that describe the unit and define its behavior. Several systemctl commands work with unit files in the background. To make finer adjustments, system administrator must edit or create unit files manually. systemd unit files locations lists three main directories where unit files are stored on the system, the /etc/systemd/system/ directory is reserved for unit files created or customized by the system administrator.

Unit file names take the following form:

```
unit_name.type_extension
```

Here, unit_name stands for the name of the unit and type_extension identifies the unit type. For a complete list of unit types, see systemd unit files.

For example, there usually is sshd.service as well as sshd.socket unit present on your system.

Unit files can be supplemented with a directory for additional configuration files. For example, to add custom configuration options to sshd.service, create the sshd.service.d/custom.conf file and insert additional directives there. For more information on configuration directories, see Modifying existing unit files.

Also, the sshd.service.wants/ and sshd.service.requires/ directories can be created. These directories contain symbolic links to unit files that are dependencies of the sshd service. The symbolic links are automatically created either during installation according to [Install] unit file options or at runtime based on [Unit] options. It is also possible to create these directories and symbolic links manually. For more details on [Install] and [Unit] options, see the tables below.

Many unit file options can be set using the so called unit specifiers – wildcard strings that are dynamically replaced with unit parameters when the unit file is loaded. This enables creation of generic unit files that serve as templates for generating instantiated units. See Working with instantiated units.

16.2. UNIT FILE STRUCTURE

Unit files typically consist of three sections:

- The [Unit] section – contains generic options that are not dependent on the type of the unit. These options provide unit description, specify the unit’s behavior, and set dependencies to other units. For a list of most frequently used [Unit] options, see Important [Unit] section options.

- The [Unit type] section – if a unit has type-specific directives, these are grouped under a section named after the unit type. For example, service unit files contain the [Service] section.
The **[Install]** section — contains information about unit installation used by `systemctl enable` and `disable` commands. For a list of options for the **[Install]** section, see [Important [Install] section options](#).

**Additional resources**

- [Important [Unit] section options](#)
- [Important [Service] section options](#)
- [Important [Install] section options](#)

### 16.3. IMPORTANT [UNIT] SECTION OPTIONS

The following tables lists important options of the [Unit] section.

<table>
<thead>
<tr>
<th>Option [a]</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
<td>A meaningful description of the unit. This text is displayed for example in the output of the <code>systemctl status</code> command.</td>
</tr>
<tr>
<td><strong>Documentation</strong></td>
<td>Provides a list of URIs referencing documentation for the unit.</td>
</tr>
<tr>
<td><strong>After</strong>[b]</td>
<td>Defines the order in which units are started. The unit starts only after the units specified in <strong>After</strong> are active. Unlike <code>Requires</code>, <strong>After</strong> does not explicitly activate the specified units. The <strong>Before</strong> option has the opposite functionality to <strong>After</strong>.</td>
</tr>
<tr>
<td><strong>Requires</strong></td>
<td>Configures dependencies on other units. The units listed in <strong>Requires</strong> are activated together with the unit. If any of the required units fail to start, the unit is not activated.</td>
</tr>
<tr>
<td><strong>Wants</strong></td>
<td>Configures weaker dependencies than <strong>Requires</strong>. If any of the listed units does not start successfully, it has no impact on the unit activation. This is the recommended way to establish custom unit dependencies.</td>
</tr>
<tr>
<td><strong>Conflicts</strong></td>
<td>Configures negative dependencies, an opposite to <strong>Requires</strong>.</td>
</tr>
</tbody>
</table>

[a] For a complete list of options configurable in the [Unit] section, see the `systemd.unit(5)` manual page.

[b] In most cases, it is sufficient to set only the ordering dependencies with **After** and **Before** unit file options. If you also set a requirement dependency with **Wants** (recommended) or **Requires**, the ordering dependency still needs to be specified. That is because ordering and requirement dependencies work independently from each other.
16.4. IMPORTANT [SERVICE] SECTION OPTIONS

The following tables lists important options of the [Service] section.

Table 16.2. Important [Service] section options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Configures the unit process startup type that affects the functionality of ExecStart and related options. One of:</td>
</tr>
<tr>
<td></td>
<td>* simple – The default value. The process started with ExecStart is the main process of the service.</td>
</tr>
<tr>
<td></td>
<td>* forking – The process started with ExecStart spawns a child process that becomes the main process of the service. The parent process exits when the startup is complete.</td>
</tr>
<tr>
<td></td>
<td>* oneshot – This type is similar to simple, but the process exits before starting consequent units.</td>
</tr>
<tr>
<td></td>
<td>* dbus – This type is similar to simple, but consequent units are started only after the main process gains a D-Bus name.</td>
</tr>
<tr>
<td></td>
<td>* notify – This type is similar to simple, but consequent units are started only after a notification message is sent via the sd_notify() function.</td>
</tr>
<tr>
<td></td>
<td>* idle – similar to simple, the actual execution of the service binary is delayed until all jobs are finished, which avoids mixing the status output with shell output of services.</td>
</tr>
<tr>
<td>ExecStart</td>
<td>Specifies commands or scripts to be executed when the unit is started. ExecStartPre and ExecStartPost specify custom commands to be executed before and after ExecStart. Type=oneshot enables specifying multiple custom commands that are then executed sequentially.</td>
</tr>
<tr>
<td>ExecStop</td>
<td>Specifies commands or scripts to be executed when the unit is stopped.</td>
</tr>
<tr>
<td>ExecReload</td>
<td>Specifies commands or scripts to be executed when the unit is reloaded.</td>
</tr>
<tr>
<td>Restart</td>
<td>With this option enabled, the service is restarted after its process exits, with the exception of a clean stop by the systemctl command.</td>
</tr>
</tbody>
</table>
RemainAfterExit

If set to True, the service is considered active even when all its processes exited. Default value is False. This option is especially useful if Type=oneshot is configured.

For a complete list of options configurable in the [Service] section, see the systemd.service(5) manual page.

16.5. IMPORTANT [INSTALL] SECTION OPTIONS

The following tables lists important options of the [Install] section.

Table 16.3. Important [Install] section options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alias</td>
<td>Provides a space-separated list of additional names for the unit. Most systemctl commands, excluding systemctl enable, can use aliases instead of the actual unit name.</td>
</tr>
<tr>
<td>RequiredBy</td>
<td>A list of units that depend on the unit. When this unit is enabled, the units listed in RequiredBy gain a Require dependency on the unit.</td>
</tr>
<tr>
<td>WantedBy</td>
<td>A list of units that weakly depend on the unit. When this unit is enabled, the units listed in WantedBy gain a Want dependency on the unit.</td>
</tr>
<tr>
<td>Also</td>
<td>Specifies a list of units to be installed or uninstalled along with the unit.</td>
</tr>
<tr>
<td>DefaultInstance</td>
<td>Limited to instantiated units, this option specifies the default instance for which the unit is enabled. See Working with instantiated units.</td>
</tr>
</tbody>
</table>

For a complete list of options configurable in the [Install] section, see the systemd.unit(5) manual page.

16.6. CREATING CUSTOM UNIT FILES

There are several use cases for creating unit files from scratch: you could run a custom daemon, create a second instance of some existing service as in Creating a custom unit file by using the second instance of the sshd service
On the other hand, if you intend just to modify or extend the behavior of an existing unit, use the instructions from Modifying existing unit files.

Procedure

The following procedure describes the general process of creating a custom service:

1. Prepare the executable file with the custom service. This can be a custom-created script, or an executable delivered by a software provider. If required, prepare a PID file to hold a constant PID for the main process of the custom service. It is also possible to include environment files to store shell variables for the service. Make sure the source script is executable (by executing the `chmod a+x`) and is not interactive.

2. Create a unit file in the `/etc/systemd/system/` directory and make sure it has correct file permissions. Execute as root:

   ```
   touch /etc/systemd/system/name.service
   chmod 664 /etc/systemd/system/name.service
   ```

   Replace `name` with a name of the service to be created. Note that file does not need to be executable.

3. Open the `name.service` file created in the previous step, and add the service configuration options. There is a variety of options that can be used depending on the type of service you wish to create, see Unit file structure.

   The following is an example unit configuration for a network-related service:

   ```
   [Unit]
   Description=service_description
   After=network.target

   [Service]
   ExecStart=path_to_executable
   Type=forking
   PIDFile=path_to_pidfile

   [Install]
   WantedBy=default.target
   ```

   Where:

   - `service_description` is an informative description that is displayed in journal log files and in the output of the `systemctl status` command.

   - the `After` setting ensures that the service is started only after the network is running. Add a space-separated list of other relevant services or targets.

   - `path_to_executable` stands for the path to the actual service executable.

   - `Type=forking` is used for daemons that make the `fork` system call. The main process of the service is created with the PID specified in `path_to_pidfile`. Find other startup types in Important [Service] section options.

   - `WantedBy` states the target or targets that the service should be started under. Think of these targets as of a replacement of the older concept of runlevels.
4. Notify **systemd** that a new **name.service** file exists by executing the following command as root:

```
systemctl daemon-reload
systemctl start name.service
```

**WARNING**
Always run the **systemctl daemon-reload** command after creating new unit files or modifying existing unit files. Otherwise, the **systemctl start** or **systemctl enable** commands could fail due to a mismatch between states of **systemd** and actual service unit files on disk. Note, that on systems with a large number of units this can take a long time, as the state of each unit has to be serialized and subsequently deserialized during the reload.

---

### 16.7. CREATING A CUSTOM UNIT FILE BY USING THE SECOND INSTANCE OF THE SSHD SERVICE

System Administrators often need to configure and run multiple instances of a service. This is done by creating copies of the original service configuration files and modifying certain parameters to avoid conflicts with the primary instance of the service. The following procedure shows how to create a second instance of the **sshd** service.

**Procedure**

1. Create a copy of the **sshd_config** file that will be used by the second daemon:

   ```
   # cp /etc/ssh/sshd{-second}.conf
   ```

2. Edit the **sshd-second_config** file created in the previous step to assign a different port number and PID file to the second daemon:

   ```
   Port 22220
   PidFile /var/run/sshd-second.pid
   ```

   See the **sshd_config**(5) manual page for more information on **Port** and **PidFile** options. Make sure the port you choose is not in use by any other service. The PID file does not have to exist before running the service, it is generated automatically on service start.

3. Create a copy of the systemd unit file for the **sshd** service:

   ```
   # cp /usr/lib/systemd/system/sshd.service /etc/systemd/system/sshd-second.service
   ```

4. Alter the **sshd-second.service** created in the previous step as follows:

   a. Modify the **Description** option:

   ```
   Description=OpenSSH server second instance daemon
   ```
b. Add sshd.service to services specified in the After option, so that the second instance starts only after the first one has already started:

```
After=syslog.target network.target auditd.service sshd.service
```

c. The first instance of sshd includes key generation, therefore remove the ExecStartPre=/usr/sbin/sshd-keygen line.

d. Add the -f /etc/ssh/sshd-second_config parameter to the sshd command, so that the alternative configuration file is used:

```
ExecStart=/usr/sbin/sshd -D -f /etc/ssh/sshd-second_config $OPTIONS
```

e. After the above modifications, the sshd-second.service should look as follows:

```
[Unit]
Description=OpenSSH server second instance daemon
After=syslog.target network.target auditd.service sshd.service

[Service]
EnvironmentFile=/etc/sysconfig/sshd
ExecStart=/usr/sbin/sshd -D -f /etc/ssh/sshd-second_config $OPTIONS
ExecReload=/bin/kill -HUP $MAINPID
KillMode=process
Restart=on-failure
RestartSec=42s

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

5. If using SELinux, add the port for the second instance of sshd to SSH ports, otherwise the second instance of sshd will be rejected to bind to the port:

```
# semanage port -a -t ssh_port_t -p tcp 22220
```

6. Enable sshd-second.service, so that it starts automatically upon boot:

```
# systemctl enable sshd-second.service
```

7. Verify if the sshd-second.service is running by using the systemctl status command.

8. Verify if the port is enabled correctly by connecting to the service:

```
$ ssh -p 22220 user@server
```

If the firewall is in use, make sure that it is configured appropriately in order to allow connections to the second instance of sshd.

### 16.8. CONVERTING SYSV INIT SCRIPTS TO UNIT FILES
Before taking time to convert a SysV init script to a unit file, make sure that the conversion was not already done elsewhere. All core services installed on Red Hat Enterprise Linux come with default unit files, and the same applies for many third-party software packages.

Converting an init script to a unit file requires analyzing the script and extracting the necessary information from it. Based on this data you can create a unit file. As init scripts can vary greatly depending on the type of the service, you might need to employ more configuration options for translation than outlined in this chapter. Note that some levels of customization that were available with init scripts are no longer supported by systemd units.

The majority of information needed for conversion is provided in the script’s header. The following example shows the opening section of the init script used to start the postfix service on Red Hat Enterprise Linux 6:

```
#!/bin/bash
# postfix      Postfix Mail Transfer Agent
# chkconfig: 2345 80 30
# description: Postfix is a Mail Transport Agent, which is the program that moves mail from one machine to another.
# processname: master
# pidfile: /var/spool/postfix/pid/master.pid
# config: /etc/postfix/main.cf
# config: /etc/postfix/master.cf
### BEGIN INIT INFO
# Provides: postfix MTA
# Required-Start: $local_fs $network $remote_fs
# Required-Stop: $local_fs $network $remote_fs
# Default-Start: 2 3 4 5
# Default-Stop: 0 1 6
# Short-Description: start and stop postfix
# Description: Postfix is a Mail Transport Agent, which is the program that moves mail from one machine to another.
### END INIT INFO
```

In the above example, only lines starting with `# chkconfig` and `# description` are mandatory, so you might not find the rest in different init files. The text enclosed between the `BEGIN INIT INFO` and `END INIT INFO` lines is called Linux Standard Base (LSB) header. If specified, LSB headers contain directives defining the service description, dependencies, and default runlevels. What follows is an overview of analytic tasks aiming to collect the data needed for a new unit file. The postfix init script is used as an example.

### 16.9. FINDING THE SYSTEMD SERVICE DESCRIPTION

You can find descriptive information about the script on the line starting with `#description`. Use this description together with the service name in the `Description` option in the `[Unit]` section of the unit file. The LSB header might contain similar data on the `#Short-Description` and `#Description` lines.

### 16.10. FINDING THE SYSTEMD SERVICE DEPENDENCIES

The LSB header might contain several directives that form dependencies between services. Most of them are translatable to systemd unit options, see the following table:

Table 16.4. Dependency options from the LSB header
<table>
<thead>
<tr>
<th>LSB Option</th>
<th>Description</th>
<th>Unit File Equivalent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Provides</td>
<td>Specifies the boot facility name of the service, that can be referenced in other init scripts (with the &quot;$&quot; prefix). This is no longer needed as unit files refer to other units by their file names.</td>
<td>–</td>
</tr>
<tr>
<td>Required-Start</td>
<td>Contains boot facility names of required services. This is translated as an ordering dependency, boot facility names are replaced with unit file names of corresponding services or targets they belong to. For example, in case of postfix, the Required-Start dependency on $network was translated to the After dependency on network.target.</td>
<td>After, Before</td>
</tr>
<tr>
<td>Should-Start</td>
<td>Constitutes weaker dependencies than Required-Start. Failed Should-Start dependencies do not affect the service startup.</td>
<td>After, Before</td>
</tr>
<tr>
<td>Required-Stop, Should-Stop</td>
<td>Constitute negative dependencies.</td>
<td>Conflicts</td>
</tr>
</tbody>
</table>

### 16.11. FINDING DEFAULT TARGETS OF THE SERVICE

The line starting with `#chkconfig` contains three numerical values. The most important is the first number that represents the default runlevels in which the service is started. Map these runlevels to equivalent systemd targets. Then list these targets in the `WantedBy` option in the [Install] section of the unit file. For example, postfix was previously started in runlevels 2, 3, 4, and 5, which translates to multi-user.target and graphical.target. Note that the graphical.target depends on multiuser.target, therefore it is not necessary to specify both. You might find information on default and forbidden runlevels also at `#Default-Start` and `#Default-Stop` lines in the LSB header.

The other two values specified on the `#chkconfig` line represent startup and shutdown priorities of the init script. These values are interpreted by `systemd` if it loads the init script, but there is no unit file equivalent.

### 16.12. FINDING FILES USED BY THE SERVICE

Init scripts require loading a function library from a dedicated directory and allow importing configuration, environment, and PID files. Environment variables are specified on the line starting with `#config` in the init script header, which translates to the `EnvironmentFile` unit file option. The PID file specified on the `#pidfile` init script line is imported to the unit file with the `PIDFile` option.

The key information that is not included in the init script header is the path to the service executable,
and potentially some other files required by the service. In previous versions of Red Hat Enterprise Linux, init scripts used a Bash case statement to define the behavior of the service on default actions, such as start, stop, or restart, as well as custom–defined actions. The following excerpt from the postfix init script shows the block of code to be executed at service start.

```bash
conf_check() {
    [ -x /usr/sbin/postfix ] || exit 5
    [ -d /etc/postfix ] || exit 6
    [ -d /var/spool/postfix ] || exit 5
}

make_aliasesdb() {
    if [ "$(/usr/sbin/postconf -h alias_database)" == "hash:/etc/aliases" ]
        then
            # /etc/aliases.db might be used by other MTA, make sure nothing
            # has touched it since our last newaliases call
            [ /etc/aliases -nt /etc/aliases.db ] ||
            [ "$ALIASESDB_STAMP" -nt /etc/aliases.db ] ||
            [ "$ALIASESDB_STAMP" -ot /etc/aliases.db ] || return
            /usr/bin/newaliases
            touch -r /etc/aliases.db "$ALIASESDB_STAMP"
        else
            /usr/bin/newaliases
        fi
    }

start() {
    [ "$EUID" != "0" ] && exit 4
    # Check that networking is up.
    [ ${NETWORKING} = "no" ] && exit 1
    conf_check
    # Start daemons.
    echo -n "$Starting postfix: 
    make_aliasesdb > /dev/null 2>&1
    [ -x $SCHROOT_UPDATE ] && $SCHROOT_UPDATE
    /usr/sbin/postfix start 2>&1
    [ "$RETRYVAL" -eq 0 ] && touch $lockfile
    echo
    return "$RETRYVAL"
}
```

The extensibility of the init script allowed specifying two custom functions, conf_check() and make_aliasesdb(), that are called from the start() function block. On closer look, several external files and directories are mentioned in the above code: the main service executable /usr/sbin/postfix, the /etc/postfix/ and /var/spool/postfix/ configuration directories, as well as the /usr/sbin/postconf/ directory.

systemd supports only the predefined actions, but enables executing custom executables with ExecStart, ExecStartPre, ExecStartPost, ExecStop, and ExecReload options. The /usr/sbin/postfix together with supporting scripts are executed on service start. Converting complex init scripts requires understanding the purpose of every statement in the script. Some of the statements are specific to the operating system version, therefore you do not need to translate them. On the other hand, some adjustments might be needed in the new environment, both in unit file as well as in the service executable and supporting files.
16.13. MODIFYING EXISTING UNIT FILES

Services installed on the system come with default unit files that are stored in the `/usr/lib/systemd/system/` directory. System Administrators should not modify these files directly, therefore any customization must be confined to configuration files in the `/etc/systemd/system/` directory.

Procedure

1. Depending on the extent of the required changes, pick one of the following approaches:

   - Create a directory for supplementary configuration files at `/etc/systemd/system/unit.d/`. This method is recommended for most use cases. It enables extending the default configuration with additional functionality, while still referring to the original unit file. Changes to the default unit introduced with a package upgrade are therefore applied automatically. See Extending the default unit configuration for more information.

   - Create a copy of the original unit file `/usr/lib/systemd/system/` in `/etc/systemd/system/` and make changes there. The copy overrides the original file, therefore changes introduced with the package update are not applied. This method is useful for making significant unit changes that should persist regardless of package updates. See Overriding the default unit configuration for details.

2. To return to the default configuration of the unit, delete custom-created configuration files in `/etc/systemd/system/`.

3. To apply changes to unit files without rebooting the system, execute:

   ```
   systemctl daemon-reload
   ```

   The `daemon-reload` option reloads all unit files and recreates the entire dependency tree, which is needed to immediately apply any change to a unit file. As an alternative, you can achieve the same result with the following command, which must be executed under the `root` user:

   ```
   init q
   ```

4. If the modified unit file belongs to a running service, this service must be restarted to accept new settings:

   ```
   systemctl restart name.service
   ```
IMPORTANT

To modify properties, such as dependencies or timeouts, of a service that is handled by a SysV initscript, do not modify the initscript itself. Instead, create a systemd drop-in configuration file for the service as described in: Extending the default unit configuration and Overriding the default unit configuration.

Then manage this service in the same way as a normal systemd service.

For example, to extend the configuration of the network service, do not modify the /etc/rc.d/init.d/network initscript file. Instead, create new directory /etc/systemd/system/network.service.d/ and a systemd drop-in file /etc/systemd/system/network.service.d/my_config.conf. Then, put the modified values into the drop-in file. Note: systemd knows the network service as network.service, which is why the created directory must be called network.service.d

16.14. EXTENDING THE DEFAULT UNIT CONFIGURATION

This section describes how to extend the default unit file with additional configuration options.

Procedure

1. To extend the default unit file with additional configuration options, first create a configuration directory in /etc/systemd/system/. If extending a service unit, execute the following command as root.

   ```bash
   mkdir /etc/systemd/system/name.service.d/
   ```

   Replace name with the name of the service you want to extend. The above syntax applies to all unit types.

2. Create a configuration file in the directory made in the previous step. Note that the file name must end with the .conf suffix. Type:

   ```bash
   touch /etc/systemd/system/name.service.d/config_name.conf
   ```

   Replace config_name with the name of the configuration file. This file adheres to the normal unit file structure, therefore all directives must be specified under appropriate sections, see Unit file structure.

   For example, to add a custom dependency, create a configuration file with the following content:

   ```ini
   [Unit]
   Requires=new_dependency
   After=new_dependency
   ```

   Where new_dependency stands for the unit to be marked as a dependency. Another example is a configuration file that restarts the service after its main process exited, with a delay of 30 seconds:

   ```ini
   [Service]
   Restart=always
   RestartSec=30
   ```
It is recommended to create small configuration files focused only on one task. Such files can be easily moved or linked to configuration directories of other services.

3. To apply changes made to the unit, execute as `root`:

```
systemctl daemon-reload
systemctl restart name.service
```

**Example 16.1. Extending the httpd.service configuration**

To modify the httpd.service unit so that a custom shell script is automatically executed when starting the Apache service, perform the following steps.

1. Create a directory and a custom configuration file:

   ```
   # mkdir /etc/systemd/system/httpd.service.d/
   # touch /etc/systemd/system/httpd.service.d/custom_script.conf
   ```

2. Provided that the script you want to start automatically with Apache is located at `/usr/local/bin/custom.sh`, insert the following text to the `custom_script.conf` file:

   ```
   [Service]
   ExecStartPost=/usr/local/bin/custom.sh
   ```

3. To apply the unit changes, execute:

   ```
   # systemctl daemon-reload
   # systemctl restart httpd.service
   ```

**NOTE**

The configuration files from configuration directories in `/etc/systemd/system/` take precedence over unit files in `/usr/lib/systemd/system/`. Therefore, if the configuration files contain an option that can be specified only once, such as `Description` or `ExecStart`, the default value of this option is overridden. Note that in the output of the `systemd-delta` command, described in Monitoring overridden units, such units are always marked as [EXTENDED], even though in sum, certain options are actually overridden.

### 16.15. OVERRIDING THE DEFAULT UNIT CONFIGURATION

This section describes how to override the default unit configuration.

**Procedure**

1. To make changes that will persist after updating the package that provides the unit file, first copy the file to the `/etc/systemd/system/` directory. To do so, execute the following command as `root`:

   ```
   cp /usr/lib/systemd/system/name.service /etc/systemd/system/name.service
   ```
Where *name* stands for the name of the service unit you wish to modify. The above syntax applies to all unit types.

2. Open the copied file with a text editor, and make the desired changes. To apply the unit changes, execute as *root*:

```bash
systemctl daemon-reload
systemctl restart name.service
```

### 16.16. CHANGING THE TIMEOUT LIMIT

You can specify a timeout value per service to prevent a malfunctioning service from freezing the system. Otherwise, timeout is set by default to 90 seconds for normal services and to 300 seconds for SysV-compatible services.

For example, to extend timeout limit for the *httpd* service:

#### Procedure

1. Copy the *httpd* unit file to the `/etc/systemd/system/` directory:

   ```bash
cp /usr/lib/systemd/system/httpd.service /etc/systemd/system/httpd.service
   ```

2. Open file `/etc/systemd/system/httpd.service` and specify the `TimeoutStartSec` value in the `[Service]` section:

   ```ini
   [Service]
   PrivateTmp=true
   TimeoutStartSec=10
   
   [Install]
   WantedBy=multi-user.target
   ```

3. Reload the *systemd* daemon:

   ```bash
   systemctl daemon-reload
   ```

4. Optional. Verify the new timeout value:

   ```bash
   systemctl show httpd -p TimeoutStartUSec
   ```

**NOTE**

To change the timeout limit globally, input the `DefaultTimeoutStartSec` in the `/etc/systemd/system.conf` file.

### 16.17. MONITORING OVERRIDDEN UNITS
This section describes how to display an overview of overridden or modified unit files.

**Procedure**

- To display an overview of overridden or modified unit files, use the following command:

  `systemd-delta`

For example, the output of the above command can look as follows:

```
[EQUIVALENT] /etc/systemd/system/default.target → /usr/lib/systemd/system/default.target
[OVERRIDDEN] /etc/systemd/system/autofs.service → /usr/lib/systemd/system/autofs.service

+ /etc/systemd/system/autofs.service 2014-11-21 10:00:58.513568275 -0500
@@ -8,7 +8,8 @@
  EnvironmentFile=-/etc/sysconfig/autofs
  ExecStart=/usr/sbin/automount $OPTIONS --pid-file /run/autofs.pid
  ExecReload=/usr/bin/kill -HUP $MAINPID
-TimeoutSec=180
+TimeoutSec=240
+Restart=Always

[Install]
WantedBy=multi-user.target

[MASKED] /etc/systemd/system/cups.service → /usr/lib/systemd/system/cups.service
[EXTENDED] /usr/lib/systemd/system/sssd.service → /etc/systemd/system/sssd.service.d/journal.conf
```

4 overridden configuration files found.

## 16.18. WORKING WITH INSTANTIATED UNITS

It is possible to instantiate multiple units from a single template configuration file at runtime. The "@" character is used to mark the template and to associate units with it. Instantiated units can be started from another unit file (using **Requires** or **Wants** options), or with the **systemctl start** command. Instantiated service units are named the following way:

```
template_name@instance_name.service
```

Where `template_name` stands for the name of the template configuration file. Replace `instance_name` with the name for the unit instance. Several instances can point to the same template file with configuration options common for all instances of the unit. Template unit name has the form of:

```
unit_name@.service
```

For example, the following **Wants** setting in a unit file:

```
Wants=getty@ttyA.service getty@ttyB.service
```
first makes systemd search for given service units. If no such units are found, the part between "@" and the type suffix is ignored and systemd searches for the `getty@.service` file, reads the configuration from it, and starts the services.

For example, the `getty@.service` template contains the following directives:

```ini
[Unit]
Description=Getty on %I
...
[Service]
ExecStart=-/sbin/agetty --noclear %I $TERM
...
```

When the `getty@ttyA.service` and `getty@ttyB.service` are instantiated from the above template, `Description` is resolved as `Getty on ttyA` and `Getty on ttyB`.

### 16.19. IMPORTANT UNIT SPECIFIERS

Wildcard characters, called unit specifiers, can be used in any unit configuration file. Unit specifiers substitute certain unit parameters and are interpreted at runtime. The following table lists unit specifiers that are particularly useful for template units.

<table>
<thead>
<tr>
<th>Unit Specifier</th>
<th>Meaning</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>%n</td>
<td>Full unit name</td>
<td>Stands for the full unit name including the type suffix. %N has the same meaning but also replaces the forbidden characters with ASCII codes.</td>
</tr>
<tr>
<td>%p</td>
<td>Prefix name</td>
<td>Stands for a unit name with type suffix removed. For instantiated units %p stands for the part of the unit name before the &quot;@&quot; character.</td>
</tr>
<tr>
<td>%i</td>
<td>Instance name</td>
<td>Is the part of the instantiated unit name between the &quot;@&quot; character and the type suffix. %I has the same meaning but also replaces the forbidden characters for ASCII codes.</td>
</tr>
<tr>
<td>%H</td>
<td>Host name</td>
<td>Stands for the hostname of the running system at the point in time the unit configuration is loaded.</td>
</tr>
</tbody>
</table>
For a complete list of unit specifiers, see the `systemd.unit(5)` manual page.

### 16.20. ADDITIONAL RESOURCES

- How to write a service unit file which enforces that particular services have to be started
- How to decide what dependencies a systemd service unit definition should have
- Is there any useful information about writing unit files?
- How to set limits for services in RHEL 7 and systemd
CHAPTER 17. OPTIMIZING SYSTEMD TO SHORTEN THE BOOT TIME

There is a list of systemd unit files that are enabled by default. System services that are defined by these unit files are automatically run at boot, which influences the boot time.

This section describes:

- The tools to examine system boot performance.
- The purpose of systemd units enabled by default, and circumstances under which you can safely disable such systemd units in order to shorten the boot time.

17.1. EXAMINING SYSTEM BOOT PERFORMANCE

To examine system boot performance, you can use the `systemd-analyze` command. This command has many options available. However, this section covers only the selected ones that may be important for systemd tuning in order to shorten the boot time.

For a complete list and detailed description of all options, see the `systemd-analyze` man page.

**Prerequisites**

Before starting to examine systemd in order to tune the boot time, you may want to list all enabled services:

**Procedure**

```
$ systemctl list-unit-files --state=enabled
```

**Analyzing overall boot time**

**Procedure**

```
$ systemd-analyze
```

**Analyzing unit initialization time**

**Procedure**

```
$ systemd-analyze blame
```

The output lists the units in descending order according to the time they took to initialize during the last successful boot.

**Identifying critical units**

**Procedure**
To identify the units that took most time to initialize at the last successful boot, use:

```bash
$ systemd-analyze critical-chain
```

The output highlights the units that critically slow down the boot with the red color.

**Figure 17.1. The output of the `systemd-analyze critical-chain` command**

```
[admin@localhost ~]$ systemd-analyze critical-chain
The time after the unit is active or started is printed after the "@" character. The time the unit takes to start is printed after the "+" character.

graphical.target @9.708s
--multi-user.target @9.708s
  -sshd.service @9.631s +397ms
  -network.target @9.614s
    -wpa_supplicant.service @10.025s +125ms
    -vnc.service @9.614s
    -basic.target @2.444s
      -sockets.target @2.444s
        -iscsi.socket @2.444s
        -sysinit.target @2.431s
          -system-update-standby.service @2.419s +10ms
          -audit.service @2.292s +12ms
          -system-updates-setup.service @2.228s +93ms
          -import-state.service @2.171s +54ms
          -local-fs.target @2.189s
            -run-user.service @9.536s
            -local-fs-pre.target @2.112s
              -lsb-localizer.service @2.087s +25ms
              -lm-event.socket @9.536s
                -system.slice
                  -slice
                  -system.slice
```

```
[admin@localhost ~]$]
```

### 17.2. A GUIDE TO SELECTING SERVICES THAT CAN BE SAFELY DISABLED

If you find the boot time of your system long, you can shorten it by disabling some of the services enabled on boot by default.

To list such services, run:

```bash
$ systemctl list-unit-files --state=enabled
```

To disable a service, run:

```bash
# systemctl disable service_name
```

However, certain services must stay enabled in order that your operating system is safe and functions in the way you need.

You can use the table below as a guide to selecting the services that you can safely disable. The table lists all services enabled by default on a minimal installation of Red Hat Enterprise Linux, and for each service it states whether this service can be safely disabled.

The table also provides more information about the circumstances under which the service can be disabled, or the reason why you should not disable the service.

**Table 17.1. Services enabled by default on a minimal installation of RHEL**

---

92
<table>
<thead>
<tr>
<th>Service name</th>
<th>Can it be disabled?</th>
<th>More information</th>
</tr>
</thead>
<tbody>
<tr>
<td>auditd.service</td>
<td>yes</td>
<td>Disable <code>auditd.service</code> only if you do not need audit messages from the kernel. Be aware that if you disable <code>auditd.service</code>, the <code>/var/log/audit/audit.log</code> file is not produced. Consequently, you are not able to retroactively review some commonly-reviewed actions or events, such as user logins, service starts or password changes. Also note that auditd has two parts: a kernel part, and a service itself. By using the <code>systemctl disable auditd</code> command, you only disable the service, but not the kernel part. To disable system auditing in its entirety, set <code>audit=0</code> on kernel command line.</td>
</tr>
<tr>
<td><a href="mailto:autovt@.service">autovt@.service</a></td>
<td>no</td>
<td>This service runs only when it is really needed, so it does not need to be disabled.</td>
</tr>
<tr>
<td>crond.service</td>
<td>yes</td>
<td>Be aware that no items from crontab will run if you disable <code>crond.service</code>.</td>
</tr>
<tr>
<td>dbus-org.fedoraproject.FirewallD1.service</td>
<td>yes</td>
<td>A symlink to <code>firewalld.service</code></td>
</tr>
<tr>
<td>dbus-org.freedesktop.NetworkManager.service</td>
<td>yes</td>
<td>A symlink to <code>NetworkManager.service</code></td>
</tr>
<tr>
<td>dbus-org.freedesktop.nm-dispatcher.service</td>
<td>yes</td>
<td>A symlink to <code>NetworkManager-dispatcher.service</code></td>
</tr>
<tr>
<td>firewalld.service</td>
<td>yes</td>
<td>Disable <code>firewalld.service</code> only if you do not need firewall.</td>
</tr>
<tr>
<td><a href="mailto:getty@.service">getty@.service</a></td>
<td>no</td>
<td>This service runs only when it is really needed, so it does not need to be disabled.</td>
</tr>
<tr>
<td>import-state.service</td>
<td>yes</td>
<td>Disable <code>import-state.service</code> only if you do not need to boot from a network storage.</td>
</tr>
<tr>
<td>irqbalance.service</td>
<td>yes</td>
<td>Disable <code>irqbalance.service</code> only if you have just one CPU. Do not disable <code>irqbalance.service</code> on systems with multiple CPUs.</td>
</tr>
<tr>
<td>Service name</td>
<td>Can it be disabled?</td>
<td>More information</td>
</tr>
<tr>
<td>---------------------------</td>
<td>---------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>kdump.service</td>
<td>yes</td>
<td>Disable <strong>kdump.service</strong> only if you do not need reports from kernel crashes.</td>
</tr>
<tr>
<td>loadmodules.service</td>
<td>yes</td>
<td>This service is not started unless the <code>/etc/rc.modules</code> or <code>/etc/sysconfig/modules</code> directory exists, which means that it is not started on a minimal RHEL installation.</td>
</tr>
<tr>
<td>lvm2-monitor.service</td>
<td>yes</td>
<td>Disable <strong>lvm2-monitor.service</strong> only if you do not use Logical Volume Manager (LVM).</td>
</tr>
<tr>
<td>microcode.service</td>
<td>no</td>
<td>Do not disable the service because it provides updates of the microcode software in CPU.</td>
</tr>
<tr>
<td>NetworkManager-dispatcher.service</td>
<td>yes</td>
<td>Disable <strong>NetworkManager-dispatcher.service</strong> only if you do not need notifications on network configuration changes (for example in static networks).</td>
</tr>
<tr>
<td>NetworkManager-wait-online.service</td>
<td>yes</td>
<td>Disable <strong>NetworkManager-wait-online.service</strong> only if you do not need working network connection available right after the boot. If the service is enabled, the system does not finish the boot before the network connection is working. This may prolong the boot time significantly.</td>
</tr>
<tr>
<td>NetworkManager.service</td>
<td>yes</td>
<td>Disable <strong>NetworkManager.service</strong> only if you do not need connection to a network.</td>
</tr>
<tr>
<td>nis-domainname.service</td>
<td>yes</td>
<td>Disable <strong>nis-domainname.service</strong> only if you do not use Network Information Service (NIS).</td>
</tr>
<tr>
<td>rhsmcertd.service</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>rngd.service</td>
<td>yes</td>
<td>Disable <strong>rngd.service</strong> only if you do not need a lot of entropy on your system, or you do not have any sort of hardware generator. Note that the service is necessary in environments that require a lot of good entropy, such as systems used for generation of X.509 certificates (for example the FreeIPA server).</td>
</tr>
<tr>
<td>rsyslog.service</td>
<td>yes</td>
<td>Disable <strong>rsyslog.service</strong> only if you do not need persistent logs, or you set <code>systemd-journald</code> to persistent mode.</td>
</tr>
<tr>
<td>Service name</td>
<td>Can it be disabled?</td>
<td>More information</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>---------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>selinux-autorelabel-mark.service</td>
<td>yes</td>
<td>Disable <code>selinux-autorelabel-mark.service</code> only if you do not use SELinux.</td>
</tr>
<tr>
<td>sshd.service</td>
<td>yes</td>
<td>Disable <code>sshd.service</code> only if you do not need remote logins by OpenSSH server.</td>
</tr>
<tr>
<td>sssd.service</td>
<td>yes</td>
<td>Disable <code>sssd.service</code> only if there are no users who log in the system over the network (for example by using LDAP or Kerberos). Red Hat recommends to disable all <code>sssd-*</code> units if you disable <code>sssd.service</code>.</td>
</tr>
<tr>
<td>syslog.service</td>
<td>yes</td>
<td>An alias for <code>rsyslog.service</code></td>
</tr>
<tr>
<td>tuned.service</td>
<td>yes</td>
<td>Disable <code>tuned.service</code> only if you do need to use performance tuning.</td>
</tr>
<tr>
<td>lvm2-lvmpolld.socket</td>
<td>yes</td>
<td>Disable <code>lvm2-lvmpolld.socket</code> only if you do not use Logical Volume Manager (LVM).</td>
</tr>
<tr>
<td>dnf-makecache.timer</td>
<td>yes</td>
<td>Disable <code>dnf-makecache.timer</code> only if you do not need your package metadata to be updated automatically.</td>
</tr>
<tr>
<td>unbound-anchor.timer</td>
<td>yes</td>
<td>Disable <code>unbound-anchor.timer</code> only if you do not need daily update of the root trust anchor for DNS Security Extensions (DNSSEC). This root trust anchor is used by Unbound resolver and resolver library for DNSSEC validation.</td>
</tr>
</tbody>
</table>

To find more information about a service, you can run one of the following commands:

- `$ systemctl cat <service_name>`
- `$ systemctl help <service_name>`

The `systemctl cat` command provides the content of the service file located under `/usr/lib/systemd/system/<service>`, as well as all applicable overrides. The applicable overrides include unit file overrides from the `/etc/systemd/system/<service>` file or drop-in files from a corresponding `unit.type.d` directory.

For more information on drop-in files, see the `systemd.unit` man page.

The `systemctl help` command shows the man page of the particular service.
17.3. ADDITIONAL RESOURCES

- `systemctl(1)` man page
- `systemd(1)` man page
- `systemd-delta(1)` man page
- `systemd.directives(7)` man page
- `systemd.unit(5)` man page
- `systemd.service(5)` man page
- `systemd.target(5)` man page
- `systemd.kill(5)` man page
- `systemd` Home Page
CHAPTER 18. INTRODUCTION TO MANAGING USER AND GROUP ACCOUNTS

The control of users and groups is a core element of Red Hat Enterprise Linux (RHEL) system administration. Each RHEL user has distinct login credentials and can be assigned to various groups to customize their system privileges.

18.1. INTRODUCTION TO USERS AND GROUPS

A user who creates a file is the owner of that file and the group owner of that file. The file is assigned separate read, write, and execute permissions for the owner, the group, and those outside that group. The file owner can be changed only by the root user. Access permissions to the file can be changed by both the root user and the file owner. A regular user can change group ownership of a file they own to a group of which they are a member of.

Each user is associated with a unique numerical identification number called user ID (UID). Each group is associated with a group ID (GID). Users within a group share the same permissions to read, write, and execute files owned by that group.

18.2. CONFIGURING RESERVED USER AND GROUP IDS

RHEL reserves user and group IDs below 1000 for system users and groups. You can find the reserved user and group IDs in the setup package. To view reserved user and group IDs, use:

```
cat /usr/share/doc/setup*/uidgid
```

It is recommended to assign IDs to the new users and groups starting at 5000, as the reserved range can increase in the future.

To make the IDs assigned to new users start at 5000 by default, modify the UID_MIN and GID_MIN parameters in the /etc/login.defs file.

Procedure

To modify and make the IDs assigned to new users start at 5000 by default:

1. Open the /etc/login.defs file in an editor of your choice.
2. Find the lines that define the minimum value for automatic UID selection.

```bash
# Min/max values for automatic uid selection in useradd
#
UID_MIN 1000
```
3. Modify the UID_MIN value to start at 5000.

```bash
# Min/max values for automatic uid selection in useradd
#
UID_MIN 5000
```
4. Find the lines that define the minimum value for automatic GID selection.
5. Modify the `GID_MIN` value to start at 5000.

```
# Min/max values for automatic gid selection in groupadd
# GID_MIN  5000
```

The dynamically assigned UIDs and GIDs for the regular users now start at 5000.

**NOTE**

The UID's and GID's of users and groups created before you changed the UID_MIN and GID_MIN values do not change.

This will allow new user's group to have same 5000+ ID as UID and GID.

**WARNING**

Do not raise IDs reserved by the system above 1000 by changing `SYS_UID_MAX` to avoid conflict with systems that retain the 1000 limit.

### 18.3. USER PRIVATE GROUPS

RHEL uses the *user private group* (**UPG**) system configuration, which makes UNIX groups easier to manage. A user private group is created whenever a new user is added to the system. The user private group has the same name as the user for which it was created and that user is the only member of the user private group.

UPGs simplify the collaboration on a project between multiple users. In addition, UPG system configuration makes it safe to set default permissions for a newly created file or directory, as it allows both the user, and the group this user is a part of, to make modifications to the file or directory.

A list of all groups is stored in the `/etc/group` configuration file.
CHAPTER 19. MANAGING USER ACCOUNTS IN THE WEB CONSOLE

The RHEL web console offers a graphical interface that enables you to execute a wide range of administrative tasks without accessing your terminal directly. For example, you can add, edit or remove system user accounts.

After reading this section, you will know:

- From where the existing accounts come from.
- How to add new accounts.
- How to set password expiration.
- How and when to terminate user sessions.

Prerequisites

- Set up the RHEL web console. For details, see Getting started using the RHEL web console.
- Log in to the RHEL web console with an account that has administrator permissions assigned. For details, see Logging in to the RHEL web console.

19.1. SYSTEM USER ACCOUNTS MANAGED IN THE WEB CONSOLE

With user accounts displayed in the RHEL web console you can:

- Authenticate users when accessing the system.
- Set the access rights to the system.

The RHEL web console displays all user accounts located in the system. Therefore, you can see at least one user account just after the first login to the web console.

After logging into the RHEL web console, you can perform the following operations:

- Create new users accounts.
- Change their parameters.
- Lock accounts.
- Terminate user sessions.

19.2. ADDING NEW ACCOUNTS USING THE WEB CONSOLE

Use the following steps for adding user accounts to the system and setting administration rights to the accounts through the RHEL web console.

Prerequisites

- The RHEL web console must be installed and accessible. For details, see Installing the web console.
Procedure

1. Log in to the RHEL web console.

2. Click **Accounts**.

3. Click **Create New Account**.

4. In the **Full Name** field, enter the full name of the user.
   The RHEL web console automatically suggests a user name from the full name and fills it in the **User Name** field. If you do not want to use the original naming convention consisting of the first letter of the first name and the whole surname, update the suggestion.

5. In the **Password/Confirm** fields, enter the password and retype it for verification that your password is correct.
   The color bar below the fields shows you the security level of the entered password, which does not allow you to create a user with a weak password.

6. Click **Create** to save the settings and close the dialog box.

7. Select the newly created account.

8. Select **Server Administrator** in the **Roles** item.

Now you can see the new account in the **Accounts** settings and you can use its credentials to connect to the system.

19.3. ENFORCING PASSWORD EXPIRATION IN THE WEB CONSOLE

By default, user accounts have set passwords to never expire. You can set system passwords to expire after a defined number of days. When the password expires, the next login attempt will prompt for a password change.

Procedure

1. Log in to the RHEL 9 web console.

2. Click **Accounts**.
3. Select the user account for which to enforce password expiration.

4. In the user account settings, click the second edit.

5. In the Password Expiration dialog box, select Require password change every ... days and enter a positive whole number representing the number of days after which the password expires.

6. Click Change.

Verification steps

- To verify that the password expiration is set, open the account settings. The RHEL 9 web console displays a link with the date of expiration.

19.4. TERMINATING USER SESSIONS IN THE WEB CONSOLE

A user creates user sessions when logging into the system. Terminating user sessions means to log the user out from the system. It can be helpful if you need to perform administrative tasks sensitive to configuration changes, for example, system upgrades.

In each user account in the RHEL 9 web console, you can terminate all sessions for the account except for the web console session you are currently using. This prevents you from losing access to your system.

Procedure

1. Log in to the RHEL 9 web console.

2. Click Accounts.

3. Click the user account for which you want to terminate the session.

4. Click Terminate Session.

   If the Terminate Session button is inactive, the user is not logged in to the system.
The RHEL web console terminates the sessions.
CHAPTER 20. MANAGING USERS FROM THE COMMAND LINE

You can manage users and groups using the command-line interface (CLI). This enables you to add, remove, and modify users and user groups in Red Hat Enterprise Linux environment.

20.1. ADDING A NEW USER FROM THE COMMAND LINE

This section describes how to use the `useradd` utility to add a new user.

**Prerequisites**

- **Root** access

**Procedure**

- To add a new user, use:

  ```
  # useradd options username
  ```

  Replace `options` with the command-line options for the `useradd` command, and replace `username` with the name of the user.

  **Example 20.1. Adding a new user**

  To add the user `sarah` with user ID `5000`, use:

  ```
  # useradd -u 5000 sarah
  ```

  **Verification steps**

  - To verify the new user is added, use the `id` utility.

    ```
    # id sarah
    ```

    The output returns:

    ```
    uid=5000(sarah) gid=5000(sarah) groups=5000(sarah)
    ```

  **Additional resources**

  - `useradd` man page

20.2. ADDING A NEW GROUP FROM THE COMMAND LINE

This section describes how to use the `groupadd` utility to add a new group.

**Prerequisites**
Procedure

To add a new group, use:

```
# groupadd options group-name
```

Replace `options` with the command-line options for the `groupadd` command, and replace `group-name` with the name of the group.

Example 20.2. Adding a new group

To add the group `sysadmins` with group ID `5000`, use:

```
# groupadd -g 5000 sysadmins
```

Verification steps

To verify the new group is added, use the `tail` utility.

```
# tail /etc/group
```

The output returns:

```
sysadmins:x:5000:
```

Additional resources

- `groupadd` man page

20.3. ADDING A USER TO A SUPPLEMENTARY GROUP FROM THE COMMAND LINE

You can add a user to a supplementary group to manage permissions or enable access to certain files or devices.

Prerequisites

- root access

Procedure

To add a group to the supplementary groups of the user, use:

```
# usermod --append -G group-name username
```

Replace `group-name` with the name of the group, and replace `username` with the name of the user.
Example 20.3. Adding a user to a supplementary group

To add the user `sysadmin` to the group `system-administrators`, use:

```
# usermod --append -G system-administrators sysadmin
```

Verification steps

- To verify the new group is added to the supplementary groups of the user `sysadmin`, use:

```
# groups sysadmin
```

The output displays:

```
sysadmin : sysadmin system-administrators
```

20.4. CREATING A GROUP DIRECTORY

Under the UPG system configuration, you can apply the `set-group identification permission` (*setgid* bit) to a directory. The *setgid* bit makes managing group projects that share a directory simpler. When you apply the *setgid* bit to a directory, files created within that directory are automatically assigned to a group that owns the directory. Any user that has the permission to write and execute within this group can now create, modify, and delete files in the directory.

The following section describes how to create group directories.

Prerequisites

- **Root** access

Procedure

1. Create a directory:

```
# mkdir directory-name
```

Replace `directory-name` with the name of the directory.

2. Create a group:

```
# groupadd group-name
```

Replace `group-name` with the name of the group.

3. Add users to the group:

```
# usermod --append -G group-name username
```

Replace `group-name` with the name of the group, and replace `username` with the name of the user.
4. Associate the user and group ownership of the directory with the `group-name` group:

```bash
# chown :group-name directory-name
```

Replace `group-name` with the name of the group, and replace `directory-name` with the name of the directory.

5. Set the write permissions to allow the users to create and modify files and directories and set the `setgid` bit to make this permission be applied within the `directory-name` directory:

```bash
# chmod g+rwxs directory-name
```

Replace `directory-name` with the name of the directory.

Now all members of the `group-name` group can create and edit files in the `directory-name` directory. Newly created files retain the group ownership of `group-name` group.

**Verification steps**

- To verify the correctness of set permissions, use:

```bash
# ls -ld directory-name
```

Replace `directory-name` with the name of the directory.

The output returns:

```
drwxrwsr-x. 2 root group-name 6 Nov 25 08:45 directory-name
```
CHAPTER 21. EDITING USER GROUPS USING THE COMMAND LINE

A user belongs to a certain set of groups that allow a logical collection of users with a similar access to files and folders. You can edit the primary and supplementary user groups from the command line to change the user’s permissions.

21.1. PRIMARY AND SUPPLEMENTARY USER GROUPS

A group is an entity which ties together multiple user accounts for a common purpose, such as granting access to particular files.

On Linux, user groups can act as primary or supplementary. Primary and supplementary groups have the following properties:

**Primary group**
- Every user has just one primary group at all times.
- You can change the user’s primary group.

**Supplementary groups**
- You can add an existing user to an existing supplementary group to manage users with the same security and access privileges within the group.
- Users can be members of zero or multiple supplementary groups.

21.2. LISTING THE PRIMARY AND SUPPLEMENTARY GROUPS OF A USER

You can list the groups of users to see which primary and supplementary groups they belong to.

**Procedure**
- Display the names of the primary and any supplementary group of a user:

  ```bash
  $ groups user-name
  ```

  Replace `user-name` with the name of the user. If you do not provide a user name, the command displays the group membership for the current user. The first group is the primary group followed by the optional supplementary groups.

  **Example 21.1. Listing of groups for user sarah:**

  ```bash
  $ groups sarah
  ```

  The output displays:

  ```
sarah : sarah wheel developer
  ```
User **sarah** has a primary group **sarah** and is a member of supplementary groups **wheel** and **developer**.

**Example 21.2. Listing of groups for user marc:**

```
$ groups marc
```

The output displays:

```
marc : marc
```

User **marc** has only a primary group **marc** and no supplementary groups.

### 21.3. CHANGING THE PRIMARY GROUP OF A USER

You can change the primary group of an existing user to a new group.

**Prerequisites:**

1. **root** access
2. The new group must exist

**Procedure**

- Change the primary group of a user:

  ```
  # usermod -g group-name user-name
  ```

  Replace **group-name** with the name of the new primary group, and replace **user-name** with the name of the user.

**NOTE**

When you change a user's primary group, the command also automatically changes the group ownership of all files in the user's home directory to the new primary group. You must fix the group ownership of files outside of the user's home directory manually.

**Example 21.3. Example of changing the primary group of a user:**

If the user **sarah** belongs to the primary group **sarah1**, and you want to change the primary group of the user to **sarah2**, use:

```
# usermod -g sarah2 sarah
```

**Verification steps**
21.4. ADDING A USER TO A SUPPLEMENTARY GROUP FROM THE COMMAND LINE

You can add a user to a supplementary group to manage permissions or enable access to certain files or devices.

**Prerequisites**

- **root** access

**Procedure**

- To add a group to the supplementary groups of the user, use:

  ```
  # usermod --append -G group-name username
  ```

  Replace `group-name` with the name of the group, and replace `username` with the name of the user.

  **Example 21.4. Adding a user to a supplementary group**

  To add the user `sysadmin` to the group `system-administrators`, use:

  ```
  # usermod --append -G system-administrators sysadmin
  ```

**Verification steps**

- To verify the new groups is added to the supplementary groups of the user `sysadmin`, use:

  ```
  # groups sysadmin
  ```

  The output displays:

  ```
  sysadmin : sysadmin system-administrators
  ```

21.5. REMOVING A USER FROM A SUPPLEMENTARY GROUP

You can remove an existing user from a supplementary group to limit their permissions or access to files and devices.

**Prerequisites**
- **root** access

**Procedure**

- Remove a user from a supplementary group:

  ```
  # gpasswd -d user-name group-name
  ```

  Replace `user-name` with the name of the user, and replace `group-name` with the name of the supplementary group.

**Example 21.5. Removing user from a supplementary group**

If the user sarah has a primary group `sarah2`, and belongs to the secondary groups `wheel` and `developers`, and you want to remove that user from the group `developers`, use:

  ```
  # gpasswd -d sarah developers
  ```

**Verification steps**

- Verify that you removed the user sarah from the secondary group developers:

  ```
  $ groups sarah
  ```

  The output displays:

  ```
  sarah : sarah2 wheel
  ```

**21.6. CHANGING ALL OF THE SUPPLEMENTARY GROUPS OF A USER**

You can overwrite the list of supplementary groups that you want the user to remain a member of.

**Prerequisites**

- **root** access
- The supplementary groups must exist

**Procedure**

- Overwrite a list of user’s supplementary groups:

  ```
  # usermod -G group-names username
  ```

  Replace `group-names` with the name of one or more supplementary groups. To add the user to several supplementary groups at once, separate the group names using commas and no intervening spaces. For example: `wheel,developer`.

  Replace `username` with the name of the user.
IMPORTANT

If the user is currently a member of a group that you do not specify, the command removes the user from the group.

Example 21.6. Changing the list of supplementary groups of a user

If the user sarah has a primary group sarah2, and belongs to the supplementary group wheel, and you want the user to belong to three more supplementary groups developer, sysadmin, and security, use:

```
# usermod -G wheel,developer,sysadmin,security sarah
```

Verification steps

- Verify that you set the list of the supplementary groups correct:

```
# groups sarah
```

The output displays:

```
sarah : sarah2 wheel developer sysadmin security
```
CHAPTER 22. MANAGING SUDO ACCESS

System administrators can grant sudo access to allow non-root users to execute administrative commands that are normally reserved for the root user. As a result, non-root users can enter such commands without logging in to the root user account.

22.1. USER AUTHORIZATIONS IN SUDOERS

The /etc/sudoers file specifies which users can run which commands using the sudo command. The rules can apply to individual users and user groups. You can also use aliases to simplify defining rules for groups of hosts, commands, and even users. Default aliases are defined in the first part of the /etc/sudoers file.

When a user tries to use sudo privileges to run a command that is not allowed in the /etc/sudoers file, the system records a message containing username: user NOT in sudoers to the journal log.

The default /etc/sudoers file provides information and examples of authorizations. You can activate a specific example rule by removing the # comment character from the beginning of the line. The authorizations section relevant for user is marked with the following introduction:

```
## Next comes the main part: which users can run what software on
## which machines (the sudoers file can be shared between multiple
## systems).
```

You can use the following format to create new sudoers authorizations and to modify existing authorizations:

```
username hostname=path/to/command
```

Where:

- `username` is the name of the user or group, for example, `user1` or `%group1`.
- `hostname` is the name of the host on which the rule applies.
- `path/to/command` is the complete absolute path to the command. You can also limit the user to only performing a command with specific options and arguments by adding those options after the command path. If you do not specify any options, the user can use the command with all options.

You can replace any of these variables with ALL to apply the rule to all users, hosts, or commands.

WARNING

With overly permissive rules, such as ALL ALL=(ALL) ALL, all users are able to run all commands as all users on all hosts. This can lead to security risks.

You can specify the arguments negatively using the ! operator. For example, use !root to specify all users except the root user. Note that using the allowlists to allow specific users, groups, and commands,
is more secure than using the blocklists to disallowing specific users, groups, and commands. By using the allowlists you also block new unauthorized users or groups.

**WARNING**

Avoid using negative rules for commands because users can overcome such rules by renaming commands using the `alias` command.

The system reads the `/etc/sudoers` file from beginning to end. Therefore, if the file contains multiple entries for a user, the entries are applied in order. In case of conflicting values, the system uses the last match, even if it is not the most specific match.

The preferred way of adding new rules to `sudoers` is by creating new files in the `/etc/sudoers.d/` directory instead of entering rules directly to the `/etc/sudoers` file. This is because the contents of this directory are preserved during system updates. In addition, it is easier to fix any errors in the separate files than in the `/etc/sudoers` file. The system reads the files in the `/etc/sudoers.d` directory when it reaches the following line in the `/etc/sudoers` file:

```
#includedir /etc/sudoers.d
```

Note that the number sign `#` at the beginning of this line is part of the syntax and does not mean the line is a comment. The names of files in that directory must not contain a period `.`, and must not end with a tilde `~`.

**Additional resources**

- `sudo(8)` and `sudoers(5)` man pages

**22.2. GRANTING SUDO ACCESS TO A USER**

System administrators can grant `sudo` access to allow non-root users to execute administrative commands. The `sudo` command provides users with administrative access without using the password of the `root` user.

When users need to perform an administrative command, they can precede that command with `sudo`. The command is then executed as if they were the `root` user.

Be aware of the following limitations:

- Only users listed in the `/etc/sudoers` configuration file can use the `sudo` command.
- The command is executed in the shell of the user, not in the `root` shell.

**Prerequisites**

- `root` access

**Procedure**

1. As root, open the `/etc/sudoers` file.
The /etc/sudoers file defines the policies applied by the sudo command.

2. In the /etc/sudoers file, find the lines that grant sudo access to users in the administrative wheel group.

```
## Allows people in group wheel to run all commands
%wheel ALL=(ALL) ALL
```

3. Make sure the line that starts with %wheel does not have the # comment character before it.

4. Save any changes, and exit the editor.

5. Add users you want to grant sudo access to into the administrative wheel group.

```
# usermod --append -G wheel <username>
```

Replace <username> with the name of the user.

**Verification steps**
- Verify that the user is added to the administrative wheel group:

```
# id <username>
uid=5000(<username>) gid=5000(<username>) groups=5000(<username>),10(wheel)
```

**Additional resources**
- sudo(8), visudo(8), and sudoers(5) man pages

### 22.3. ENABLING UNPRIVILEGED USERS TO RUN CERTAIN COMMANDS

As an administrator, you can allow unprivileged users to enter certain commands on specific workstations by configuring a policy in the /etc/sudoers.d/ directory.

For example, you can enable the user <example.user> to install programs on the host.example.com workstation using the dnf command with sudo privileges.

**Prerequisites**
- You must have root access to the system.

**Procedure**
1. As root, create a new sudoers.d directory under /etc:

```
# mkdir -p /etc/sudoers.d/
```

2. Create a new file in the /etc/sudoers.d directory:
# visudo -f /etc/sudoers.d/<example.user>

The file opens automatically.

3. Add the following line to the /etc/sudoers.d/<example.user> file:

```bash
<example.user> <host.example.com> = /usr/bin/dnf
```

To allow two and more commands on the same host on one line, you can list them separated by a comma followed by a space.

4. Optional: To receive email notifications every time the user <example.user> attempts to use sudo privileges, add the following lines to the file:

```bash
Defaults mail_always
Defaults mailto="<email@example.com>"
```

5. Save the changes, and exit the editor.

Verification

1. To verify if the user <example.user> can run the dnf command with sudo privileges, switch the account:

```bash
# su <example.user> -
```

2. Enter the sudo dnf command:

```
$ sudo dnf
[sudo] password for <example.user>:
```

Enter the sudo password for the user <example.user>.

3. The system displays the list of dnf commands and options:

```
... usage: dnf [options] COMMAND ...
```

If the system returns the <example.user> is not in the sudoers file. This incident will be reported error message, you have not created the file for <example.user> in /etc/sudoers.d/.

If you receive the <example.user> is not allowed to run sudo on <host.example.com> error message, you have not completed the configuration correctly. Ensure that you are logged in as root and that you followed the steps thoroughly.

Additional resources

- sudo(8), visudo(8), and sudoers(5) man pages
CHAPTER 23. CHANGING AND RESETTING THE ROOT PASSWORD

If the existing root password is no longer satisfactory or is forgotten, you can change or reset it both as the root user and a non-root user.

23.1. CHANGING THE ROOT PASSWORD AS THE ROOT USER

This section describes how to use the `passwd` command to change the root password as the root user.

**Prerequisites**

- Root access

**Procedure**

- To change the root password, use:

  ```
  # passwd
  ```

  You are prompted to enter your current password before you can change it.

23.2. CHANGING OR RESETTING THE FORGOTTEN ROOT PASSWORD AS A NON-ROOT USER

This section describes how to use the `passwd` command to change or reset the forgotten root password as a non-root user.

**Prerequisites**

- You are able to log in as a non-root user.
- You are a member of the administrative wheel group.

**Procedure**

- To change or reset the root password as a non-root user that belongs to the wheel group, use:

  ```
  $ sudo passwd root
  ```

  You are prompted to enter your current non-root password before you can change the root password.

23.3. RESETTING THE ROOT PASSWORD ON BOOT

If you are unable to log in as a non-root user or do not belong to the administrative wheel group, you can reset the root password on boot by switching into a specialized chroot jail environment.

**Procedure**

1. Reboot the system and, on the GRUB 2 boot screen, press the e key to interrupt the boot process.
The kernel boot parameters appear.

```bash
load_video
set gfxpayload=keep
insmod gzio
linux ($root)/vmlinuz-5.14.0-70.22.1.e19_0.x86_64 root=/dev/mapper/rhel-root ro crash
kernel=auto resume=/dev/mapper/rhel-swap rd.lvm.lv/swap rhgb quiet
initrd ($root)/initramfs-5.14.0-70.22.1.e19_0.x86_64.img $tuned_initrd
```

2. Go to the end of the line that starts with `linux`.

```bash
linux ($root)/vmlinuz-5.14.0-70.22.1.e19_0.x86_64 root=/dev/mapper/rhel-root ro crash
kernel=auto resume=/dev/mapper/rhel-swap rd.lvm.lv/swap rhgb quiet
```

Press `Ctrl+e` to jump to the end of the line.

3. Add `rd.break` to the end of the line that starts with `linux`.

```bash
linux ($root)/vmlinuz-5.14.0-70.22.1.e19_0.x86_64 root=/dev/mapper/rhel-root ro crash
kernel=auto resume=/dev/mapper/rhel-swap rd.lvm.lv/swap rhgb quiet rd.break
```

4. Press `Ctrl+x` to start the system with the changed parameters.

   The `switch_root` prompt appears.

5. Remount the file system as writable:

   ```bash
   mount -o remount,rw /sysroot
   ```

   The file system is mounted as read-only in the `/sysroot` directory. Remounting the file system as writable allows you to change the password.

6. Enter the `chroot` environment:

   ```bash
   chroot /sysroot
   ```

   The `sh-4.4#` prompt appears.

7. Reset the `root` password:

   ```bash
   passwd
   ```

   Follow the instructions displayed by the command line to finalize the change of the `root` password.

8. Enable the SELinux relabeling process on the next system boot:

   ```bash
   touch /.autorelabel
   ```

9. Exit the `chroot` environment:

   ```bash
   exit
   ```

10. Exit the `switch_root` prompt:
exit

11. Wait until the SELinux relabeling process is finished. Note that relabeling a large disk might take a long time. The system reboots automatically when the process is complete.

**Verification steps**

1. To verify that the root password is successfully changed, log in as a normal user and open the Terminal.

2. Run the interactive shell as root:

   ```bash
   $ su
   ```

3. Enter your new root password.

4. Print the user name associated with the current effective user ID:

   ```bash
   whoami
   ```

   The output returns:

   ```bash
   root
   ```
CHAPTER 24. MANAGING FILE PERMISSIONS

File permissions control the ability of user and group accounts to view, modify, access, and execute the contents of the files and directories.

Every file or directory has three levels of ownership:

- User owner (u).
- Group owner (g).
- Others (o).

Each level of ownership can be assigned the following permissions:

- Read (r).
- Write (w).
- Execute (x).

Note that the execute permission for a file allows you to execute that file. The execute permission for a directory allows you to access the contents of the directory, but not execute it.

When a new file or directory is created, the default set of permissions are automatically assigned to it. The default permissions for a file or directory are based on two factors:

- Base permission.
- The *user file-creation mode mask* (umask).

24.1. BASE FILE PERMISSIONS

Whenever a new file or directory is created, a base permission is automatically assigned to it. Base permissions for a file or directory can be expressed in *symbolic* or *octal* values.

<table>
<thead>
<tr>
<th>Permission</th>
<th>Symbolic value</th>
<th>Octal value</th>
</tr>
</thead>
<tbody>
<tr>
<td>No permission</td>
<td>---</td>
<td>0</td>
</tr>
<tr>
<td>Execute</td>
<td>--x</td>
<td>1</td>
</tr>
<tr>
<td>Write</td>
<td>-w-</td>
<td>2</td>
</tr>
<tr>
<td>Write and execute</td>
<td>-wx</td>
<td>3</td>
</tr>
<tr>
<td>Read</td>
<td>r--</td>
<td>4</td>
</tr>
<tr>
<td>Read and execute</td>
<td>r-x</td>
<td>5</td>
</tr>
<tr>
<td>Read and write</td>
<td>rw-</td>
<td>6</td>
</tr>
</tbody>
</table>
The base permission for a directory is **777** (drwxrwxrwx), which grants everyone the permissions to read, write, and execute. This means that the directory owner, the group, and others can list the contents of the directory, create, delete, and edit items within the directory, and descend into it.

Note that individual files within a directory can have their own permission that might prevent you from editing them, despite having unrestricted access to the directory.

The base permission for a file is **666** (-rw-rw-rw-), which grants everyone the permissions to read and write. This means that the file owner, the group, and others can read and edit the file.

**Example 24.1. Permissions for a file**

If a file has the following permissions:

```bash
$ ls -l
-rwxrw----. 1 sysadmins sysadmins 2 Mar 2 08:43 file
```

- `-` indicates it is a file.
- **rwx** indicates that the file owner has permissions to read, write, and execute the file.
- **rw-** indicates that the group has permissions to read and write, but not execute the file.
- `---` indicates that other users have no permission to read, write, or execute the file.
- `. ` indicates that the SELinux security context is set for the file.

**Example 24.2. Permissions for a directory**

If a directory has the following permissions:

```bash
$ ls -dl
drwxr-----. 1 sysadmins sysadmins 2 Mar 2 08:43 directory
```

- `d` indicates it is a directory.
- **rwx** indicates that the directory owner has the permissions to read, write, and access the contents of the directory.
  
  As a directory owner, you can list the items (files, subdirectories) within the directory, access the content of those items, and modify them.

- `r--` indicates that the group has permissions to read, but not write or access the contents of the directory.
  
  As a member of the group that owns the directory, you can list the items within the directory. You cannot access information about the items within the directory or modify them.

- `---` indicates that other users have no permission to read, write, or access the contents of the directory.
  
  As someone who is not a user owner, or as group owner of the directory, you cannot list the items within the directory, access information about those items, or modify them.
indicates that the SELinux security context is set for the directory.

NOTE
The base permission that is automatically assigned to a file or directory is not the default permission the file or directory ends up with. When you create a file or directory, the base permission is altered by the umask. The combination of the base permission and the umask creates the default permission for files and directories.

24.2. USER FILE-CREATION MODE MASK

The user file-creation mode mask (umask) is a variable that controls how file permissions are set for newly created files and directories. The umask automatically removes permissions from the base permission value to increase the overall security of a Linux system. The umask can be expressed in symbolic or octal values.

<table>
<thead>
<tr>
<th>Permission</th>
<th>Symbolic value</th>
<th>Octal value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read, write, and execute</td>
<td>rwx</td>
<td>0</td>
</tr>
<tr>
<td>Read and write</td>
<td>rw-</td>
<td>1</td>
</tr>
<tr>
<td>Read and execute</td>
<td>r-x</td>
<td>2</td>
</tr>
<tr>
<td>Read</td>
<td>r--</td>
<td>3</td>
</tr>
<tr>
<td>Write and execute</td>
<td>-wx</td>
<td>4</td>
</tr>
<tr>
<td>Write</td>
<td>-w-</td>
<td>5</td>
</tr>
<tr>
<td>Execute</td>
<td>--x</td>
<td>6</td>
</tr>
<tr>
<td>No permissions</td>
<td>---</td>
<td>7</td>
</tr>
</tbody>
</table>

The default umask for a standard user is 0002. The default umask for a root user is 0022.

The first digit of the umask represents special permissions (sticky bit, ). The last three digits of the umask represent the permissions that are removed from the user owner (u), group owner (g), and others (o) respectively.

Example 24.3. Applying the umask when creating a file

The following example illustrates how the umask with an octal value of 0137 is applied to the file with the base permission of 777, to create the file with the default permission of 640.
24.3. DEFAULT FILE PERMISSIONS

The default permissions are set automatically for all newly created files and directories. The value of the default permissions is determined by applying the \textit{umask} to the base permission.

\textbf{Example 24.4. Default permissions for a directory created by a standard user}

When a \textbf{standard user} creates a new \textbf{directory}, the \textit{umask} is set to 002 (rwxrwxr-x), and the base permissions for a directory are set to 777 (rwxrwxrwx). This brings the default permissions to 775 (drwxrwxr-x).

<table>
<thead>
<tr>
<th></th>
<th>Symbolic value</th>
<th>Octal value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base permission</td>
<td>rwxrwxrwx</td>
<td>777</td>
</tr>
<tr>
<td>Umask</td>
<td>rwxrwxr-x</td>
<td>002</td>
</tr>
<tr>
<td>Default permission</td>
<td>rwxrwxr-x</td>
<td>775</td>
</tr>
</tbody>
</table>

This means that the directory owner and the group can list the contents of the directory, create, delete, and edit items within the directory, and descend into it. Other users can only list the contents of the directory and descend into it.

\textbf{Example 24.5. Default permissions for a file created by a standard user}

When a \textbf{standard user} creates a new \textbf{file}, the \textit{umask} is set to 002 (rwxrwxr-x), and the base permissions for a file are set to 666 (rw-rw-rw-). This brings the default permissions to 664 (-rw-rw-r-).
Symbolic value | Octal value
---|---
**Base permission** | rw-rw-rw-
**Umask** | rwxrwxr-x
**Default permission** | rw-rw-r--

This means that the file owner and the group can read and edit the file, while other users can only read the file.

**Example 24.6. Default permissions for a directory created by the root user**

When a *root user* creates a new *directory*, the *umask* is set to 022 (rwxr-xr-x), and the base permissions for a directory are set to 777 (rwxrwxrwx). This brings the default permissions to 755 (rwxr-xr-x).

Symbolic value | Octal value
---|---
**Base permission** | rwxrwxrwx
**Umask** | rwxr-xr-x
**Default permission** | rwxr-xr-x

This means that the directory owner can list the contents of the directory, create, delete, and edit items within the directory, and descend into it. The group and others can only list the contents of the directory and descend into it.

**Example 24.7. Default permissions for a file created by the root user**

When a *root user* creates a new *file*, the *umask* is set to 022 (rwxr-xr-x), and the base permissions for a file are set to 666 (rw-rw-rw-). This brings the default permissions to 644 (-rw-r--).

Symbolic value | Octal value
---|---
**Base permission** | rw-rw-rw-
**Umask** | rwxr-xr-x
**Default permission** | rw-r--

This means that the file owner can read and edit the file, while the group and others can only read the file.
NOTE

For security reasons, regular files cannot have execute permissions by default, even if the umask is set to 000 (rwxrwxrwx). However, directories can be created with execute permissions.

24.4. CHANGING FILE PERMISSIONS USING SYMBOLIC VALUES

You can use the chmod utility with symbolic values (a combination of letters and signs) to change file permissions for a file or directory.

You can assign the following permissions:

- Read (r)
- Write (w)
- Execute (x)

Permissions can be assigned to the following levels of ownership:

- User owner (u)
- Group owner (g)
- Other (o)
- All (a)

To add or remove permissions you can use the following signs:

- + to add the permissions on top of the existing permissions
- - to remove the permissions from the existing permission
- = to remove the existing permissions and explicitly define the new ones

Procedure

- To change the permissions for a file or directory, use:

  ```
  $ chmod <level><operation><permission> file-name
  ```

  Replace `<level>` with the level of ownership you want to set the permissions for. Replace `<operation>` with one of the signs. Replace `<permission>` with the permissions you want to assign. Replace `file-name` with the name of the file or directory. For example, to grant everyone the permissions to read, write, and execute (rwx) `my-script.sh`, use the `chmod a=rwx my-script.sh` command.

  See Base file permissions for more details.

Verification steps

- To see the permissions for a particular file, use:

  ```
  $ ls -l file-name
  ```
Replace `file-name` with the name of the file.

- To see the permissions for a particular directory, use:
  
  ```bash
  $ ls -dl directory-name
  
  Replace `directory-name` with the name of the directory.
  
  - To see the permissions for all the files within a particular directory, use:
  
  ```bash
  $ ls -l directory-name
  
  Replace `directory-name` with the name of the directory.

Example 24.8. Changing permissions for files and directories

- To change file permissions for `my-file.txt` from `-rw-rw-r--` to `-rw------`, use:
  1. Display the current permissions for `my-file.txt`:

  ```bash
  $ ls -l my-file.txt
  -rw-rw-r--. 1 username username 0 Feb 24 17:56 my-file.txt
  
  2. Remove the permissions to read, write, and execute (`rwx`) the file from group owner (`g`) and others (`o`):

  ```bash
  $ chmod go= my-file.txt
  
  Note that any permission that is not specified after the equals sign (=`) is automatically prohibited.
  
  3. Verify that the permissions for `my-file.txt` were set correctly:

  ```bash
  $ ls -l my-file.txt
  -rw-------. 1 username username 0 Feb 24 17:56 my-file.txt
  
  - To change file permissions for `my-directory` from `drwxrwx---` to `drwxrwxr-x`, use:
  1. Display the current permissions for `my-directory`:

  ```bash
  $ ls -dl my-directory
  drwxrwx---. 2 username username 4096 Feb 24 18:12 my-directory
  
  2. Add the read and execute (`r-x`) access for all users (`a`):

  ```bash
  $ chmod o+rx my-directory
  
  3. Verify that the permissions for `my-directory` and its content were set correctly:

  ```bash
  $ ls -dl my-directory
  drwxrwxr-x. 2 username username 4096 Feb 24 18:12 my-directory
24.5. CHANGING FILE PERMISSIONS USING OCTAL VALUES

You can use the `chmod` utility with octal values (numbers) to change file permissions for a file or directory.

**Procedure**

- To change the file permissions for an existing file or directory, use:

  ```
  $ chmod octal_value file-name
  ```

  Replace `file-name` with the name of the file or directory. Replace `octal_value` with an octal value. See [Base file permissions](#) for more details.
CHAPTER 25. MANAGING THE UMASK

You can use the `umask` utility to display, set, or change the current or default value of the `umask`.

25.1. DISPLAYING THE CURRENT VALUE OF THE UMASK

You can use the `umask` utility to display the current value of the `umask` in symbolic or octal mode.

**Procedure**

- To display the current value of the `umask` in symbolic mode, use:
  `$ umask -S`

- To display the current value of the `umask` in the octal mode, use:
  `$ umask`

**NOTE**

When displaying the `umask` in octal mode, you may notice it displayed as a four digit number (`0002` or `0022`). The first digit of the `umask` represents a special bit (sticky bit, SGID bit, or SUID bit). If the first digit is set to 0, the special bit is not set.

25.2. DISPLAYING THE DEFAULT BASH UMASK

There are a number of shells you can use, such as `bash`, `ksh`, `zsh` and `tcsh`. Those shells can behave as login or non-login shells. You can invoke the login shell by opening a native or a GUI terminal.

To determine whether you are executing a command in a login or a non-login shell, use the `echo $0` command.

**Example 25.1. Determining if you are working in a login or a non-login bash shell**

- If the output of the `echo $0` command returns `bash`, you are executing the command in a non-login shell.
  ```
  $ echo $0
  bash
  ```

  The default `umask` for the non-login shell is set in the `/etc/bashrc` configuration file.

- If the output of the `echo $0` command returns `-bash`, you are executing the command in a login shell.
  ```
  # echo $0
  -bash
  ```

  The default `umask` for the login shell is set in the `/etc/login.defs` configuration file.
Procedure

- To display the default **bash umask** for the non-login shell, use:

  ```bash
  $ grep umask /etc/bashrc
  ```

  The output returns:

  ```
  # By default, we want umask to get set. This sets it for non-login shell.
  umask 002
  umask 022
  ```

- To display the default **bash umask** for the login shell, use:

  ```bash
  grep "UMASK" /etc/login.defs
  ```

  The output returns:

  ```
  # UMASK is also used by useradd(8) and newusers(8) to set the mode for new
  UMASK 022
  # If HOME_MODE is not set, the value of UMASK is used to create the mode.
  ```

### 25.3. SETTING THE UMASK USING SYMBOLIC VALUES

You can use the **umask** utility with symbolic values (a combination letters and signs) to set the **umask** for the current shell session.

You can assign the following **permissions**:

- Read (r)
- Write (w)
- Execute (x)

Permissions can be assigned to the following **levels of ownership**:

- User owner (u)
- Group owner (g)
- Other (o)
- All (a)

To add or remove permissions you can use the following **signs**:

- `+` to add the permissions on top of the existing permissions
- `-` to remove the permissions from the existing permission
- `=` to remove the existing permissions and explicitly define the new ones
NOTE
Any permission that is not specified after the equals sign (=) is automatically prohibited.

Procedure

- To set the umask for the current shell session, use:

  $ umask -S <level><operation><permission>

Replace <level> with the level of ownership you want to set the umask for. Replace <operation> with one of the signs. Replace <permission> with the permissions you want to assign. For example, to set the umask to u=rwx,g=rwx,o=rwx, use umask -S a=rwx.

See User file-creation mode for more details.

NOTE
The umask is only valid for the current shell session.

25.4. SETTING THE UMASK USING OCTAL VALUES
You can use the umask utility with octal values (numbers) to set the umask for the current shell session.

Procedure

- To set the umask for the current shell session, use:

  $ umask octal_value

Replace octal_value with an octal value. See User file-creation mode mask for more details.

NOTE
The umask is only valid for the current shell session.

25.5. CHANGING THE DEFAULT UMASK FOR THE NON-LOGIN SHELL
You can change the default bash umask for standard users by modifying the /etc/bashrc file.

Prerequisites

- root access

Procedure

1. As root, open the /etc/bashrc file in the editor.

2. Modify the following sections to set a new default bash umask:

   if [ $UID -gt 199 ] && [ “id -gn” = “id -un” ]; then
     umask 002
else
  umask 022
fi

Replace the default octal value of the umask (022) with another octal value. See User file-creation mode mask for more details.

3. Save the changes and exit the editor.

25.6. CHANGING THE DEFAULT UMASK FOR THE LOGIN SHELL

You can change the default bash umask for the root user by modifying the /etc/login.defs file.

Prerequisites

- root access

Procedure

1. As root, open the /etc/login.defs file in the editor.

2. Modify the following sections to set a new default bash umask:

   # Default initial "umask" value used by login(1) on non-PAM enabled systems.
   # Default "umask" value for pam_umask(8) on PAM enabled systems.
   # UMASK is also used by useradd(8) and newusers(8) to set the mode for new
   # home directories if HOME_MODE is not set.
   # 022 is the default value, but 027, or even 077, could be considered
   # for increased privacy. There is no One True Answer here: each sysadmin
   # must make up their mind.

   UMASK    022

Replace the default octal value of the umask (022) with another octal value. See User file-creation mode mask for more details.

3. Save the changes and exit the editor.

25.7. CHANGING THE DEFAULT UMASK FOR A SPECIFIC USER

You can change the default umask for a specific user by modifying the .bashrc for that user.

Procedure

- Append the line that specifies the octal value of the umask into the .bashrc file for the particular user.

  $ echo 'umask octal_value' >> /home/username/.bashrc

Replace octal_value with an octal value and replace username with the name of the user. See User file-creation mode mask for more details.
25.8. SETTING DEFAULT PERMISSIONS FOR NEWLY CREATED HOME DIRECTORIES

You can change the permission modes for home directories of newly created users by modifying the /etc/login.defs file.

Procedure

1. As root, open the /etc/login.defs file in the editor.

2. Modify the following section to set a new default HOME_MODE:

   ```
   # HOME_MODE is used by useradd(8) and newusers(8) to set the mode for new # home directories.
   # If HOME_MODE is not set, the value of UMASK is used to create the mode.
   HOME_MODE       0700
   ```

   Replace the default octal value (0700) with another octal value. The selected mode will be used to create the permissions for the home directory.

3. If HOME_MODE is set, save the changes and exit the editor.

4. If HOME_MODE is not set, modify the UMASK to set the mode for the newly created home directories:

   ```
   # Default initial "umask" value used by login(1) on non-PAM enabled systems.
   # Default "umask" value for pam_umask(8) on PAM enabled systems.
   # UMASK is also used by useradd(8) and newusers(8) to set the mode for new # home directories if HOME_MODE is not set.
   # 022 is the default value, but 027, or even 077, could be considered
   # for increased privacy. There is no One True Answer here: each sysadmin # must make up their mind.
   UMASK           022
   ```

   Replace the default octal value (022) with another octal value. See User file-creation mode mask for more details.

5. Save the changes and exit the editor.
CHAPTER 26. MANAGING THE ACCESS CONTROL LIST

Each file and directory can only have one user owner and one group owner at a time. If you want to grant a user permissions to access specific files or directories that belong to a different user or group while keeping other files and directories private, you can utilize Linux Access Control Lists (ACLs).

26.1. DISPLAYING THE CURRENT ACCESS CONTROL LIST

You can use the `getfacl` utility to display the current ACL.

Procedure

- To display the current ACL for a particular file or directory, use:

  ```bash
  $ getfacl file-name
  ```

  Replace `file-name` with the name of the file or directory.

26.2. SETTING THE ACCESS CONTROL LIST

You can use the `setfacl` utility to set the ACL for a file or directory.

Prerequisites

- root access.

Procedure

- To set the ACL for a file or directory, use:

  ```bash
  # setfacl -m u:username:symbolic_value file-name
  ```

  Replace `username` with the name of the user, `symbolic_value` with a symbolic value, and `file-name` with the name of the file or directory. For more information see the `setfacl` man page.

Example 26.1. Modifying permissions for a group project

The following example describes how to modify permissions for the `group-project` file owned by the root user that belongs to the root group so that this file is:

- Not executable by anyone.
- The user `andrew` has the `rw-` permissions.
- The user `susan` has the `---` permissions.
- Other users have the `r--` permissions.

Procedure

```bash
# setfacl -m u:andrew:rw- group-project
# setfacl -m u:susan:--- group-project
```
Verification steps

- To verify that the user `andrew` has the `rw-` permission, the user `susan` has the `---` permission, and other users have the `r--` permission, use:

  ```bash
  $ getfacl group-project
  
  The output returns:
  
  # file: group-project
  # owner: root
  # group: root
  user:andrew:rw-
  user:susan:---
  group::r--
  mask::rw-
  other::r--
  ```
CHAPTER 27. USING THE CHRONY SUITE TO CONFIGURE NTP

Accurate timekeeping is important for a number of reasons in IT. In networking for example, accurate time stamps in packets and logs are required. In Linux systems, the NTP protocol is implemented by a daemon running in user space.

The user space daemon updates the system clock running in the kernel. The system clock can keep time by using various clock sources. Usually, the Time Stamp Counter (TSC) is used. The TSC is a CPU register which counts the number of cycles since it was last reset. It is very fast, has a high resolution, and there are no interruptions.

Starting with Red Hat Enterprise Linux 8, the NTP protocol is implemented by the chronyd daemon, available from the repositories in the chrony package.

The following sections describe how to use the chrony suite to configure NTP.

27.1. INTRODUCTION TO CHRONY SUITE

chrony is an implementation of the Network Time Protocol (NTP). You can use chrony:

- To synchronize the system clock with NTP servers
- To synchronize the system clock with a reference clock, for example a GPS receiver
- To synchronize the system clock with a manual time input
- As an NTPv4(RFC 5905) server or peer to provide a time service to other computers in the network

chrony performs well in a wide range of conditions, including intermittent network connections, heavily congested networks, changing temperatures (ordinary computer clocks are sensitive to temperature), and systems that do not run continuously, or run on a virtual machine.

Typical accuracy between two machines synchronized over the Internet is within a few milliseconds, and for machines on a LAN within tens of microseconds. Hardware timestamping or a hardware reference clock may improve accuracy between two machines synchronized to a sub-microsecond level.

chrony consists of chronyd, a daemon that runs in user space, and chronyc, a command line program which can be used to monitor the performance of chronyd and to change various operating parameters when it is running.

The chrony daemon, chronyd, can be monitored and controlled by the command line utility chronyc. This utility provides a command prompt which allows entering a number of commands to query the current state of chronyd and make changes to its configuration. By default, chronyd accepts only commands from a local instance of chronyc, but it can be configured to accept monitoring commands also from remote hosts. The remote access should be restricted.

27.2. USING CHRONYC TO CONTROL CHRONYD

This section describes how to control chronyd using the chronyc command line utility.

Procedure
1. To make changes to the local instance of `chronyd` using the command line utility `chronyc` in interactive mode, enter the following command as `root`:

```
# chronyc
```

`chronyc` must run as `root` if some of the restricted commands are to be used.

The `chronyc` command prompt will be displayed as follows:

```
chronyc>
```

2. To list all of the commands, type `help`.

3. Alternatively, the utility can also be invoked in non-interactive command mode if called together with a command as follows:

```
chronyc command
```

**NOTE**

Changes made using `chronyc` are not permanent, they will be lost after a `chronyd` restart. For permanent changes, modify `/etc/chrony.conf`. 
CHAPTER 28. USING CHRONY

The following sections describe how to install, start, and stop **chronyd**, and how to check if **chrony** is synchronized. Sections also describe how to manually adjust System Clock.

### 28.1. MANAGING CHRONY

The following procedure describes how to install, start, stop, and check the status of **chronyd**.

**Procedure**

1. The **chrony** suite is installed by default on Red Hat Enterprise Linux. To ensure that it is, run the following command as **root**:

   ```
   # dnf install chrony
   ```

   The default location for the **chrony** daemon is `/usr/sbin/chronyd`. The command line utility will be installed to `/usr/bin/chronyc`.

2. To check the status of **chronyd**, issue the following command:

   ```
   $ systemctl status chronyd
   chronyd.service - NTP client/server
   Loaded: loaded (/usr/lib/systemd/system/chronyd.service; enabled)
   Active: active (running) since Wed 2013-06-12 22:23:16 CEST; 11h ago
   ```

3. To start **chronyd**, issue the following command as **root**:

   ```
   # systemctl start chronyd
   ```

   To ensure **chronyd** starts automatically at system start, issue the following command as **root**:

   ```
   # systemctl enable chronyd
   ```

4. To stop **chronyd**, issue the following command as **root**:

   ```
   # systemctl stop chronyd
   ```

   To prevent **chronyd** from starting automatically at system start, issue the following command as **root**:

   ```
   # systemctl disable chronyd
   ```

### 28.2. CHECKING IF CHRONY IS SYNCHRONIZED

The following procedure describes how to check if **chrony** is synchronized with the use of the **tracking**, **sources**, and **sourcestats** commands.

**Procedure**

1. To check **chrony** tracking, issue the following command:
2. The `sources` command displays information about the current time sources that `chronyd` is accessing. To check `chrony` sources, issue the following command:

```bash
$ chronyc sources
210 Number of sources = 3
MS Name/IP address Stratum Poll Reach LastRx Last sample
===========================================================================
# GPS0                          0   4   377    11   -479ns[-621ns] /-  134ns
^? a.b.c                         2   6   377    23   -923us[-924us] +/-   43ms
^ d.e.f                         1   6   377    21  -2629us[-2619us] +/-   86ms
```

The optional argument `-v` can be specified, meaning verbose. In this case, extra caption lines are shown as a reminder of the meanings of the columns.

3. The `sourcestats` command displays information about the drift rate and offset estimation process for each of the sources currently being examined by `chronyd`. To check `chrony` source statistics, issue the following command:

```bash
$ chronyc sourcestats
210 Number of sources = 1
Name/IP Address NP NR Span Frequency Freq Skew Offset Std Dev
===========================================================================
====
abc.def.ghi                11   5   46m     -0.001      0.045      1us    25us
```

The optional argument `-v` can be specified, meaning verbose. In this case, extra caption lines are shown as a reminder of the meanings of the columns.

Additional resources

- `chronyc(1)` man page

### 28.3. MANUALLY ADJUSTING THE SYSTEM CLOCK

The following procedure describes how to manually adjust the System Clock.

**Procedure**
To step the system clock immediately, bypassing any adjustments in progress by slewing, issue the following command as root:

```
# chronyc makestep
```

If the `rtcfile` directive is used, the real-time clock should not be manually adjusted. Random adjustments would interfere with `chrony`’s need to measure the rate at which the real-time clock drifts.

### 28.4. SETTING UP CHRONY FOR A SYSTEM IN AN ISOLATED NETWORK

For a network that is never connected to the Internet, one computer is selected to be the master timeserver. The other computers are either direct clients of the master, or clients of clients. On the master, the drift file must be manually set with the average rate of drift of the system clock. If the master is rebooted, it will obtain the time from surrounding systems and calculate an average to set its system clock. Thereafter it resumes applying adjustments based on the drift file. The drift file will be updated automatically when the `settime` command is used.

The following procedure describes how to set up `chrony` for a system in an isolated network.

**Procedure**

1. On the system selected to be the master, using a text editor running as root, edit `/etc/chrony.conf` as follows:

   ```
   driftfile /var/lib/chrony/drift
   commandkey 1
   keyfile /etc/chrony.keys
   initstepslew 10 client1 client3 client6
   local stratum 8
   manual
   allow 192.0.2.0
   ```

   Where **192.0.2.0** is the network or subnet address from which the clients are allowed to connect.

2. On the systems selected to be direct clients of the master, using a text editor running as root, edit the `/etc/chrony.conf` as follows:

   ```
   server master
   driftfile /var/lib/chrony/drift
   logdir /var/log/chrony
   log measurements statistics tracking
   keyfile /etc/chrony.keys
   commandkey 24
   local stratum 10
   initstepslew 20 master
   allow 192.0.2.123
   ```

   Where **192.0.2.123** is the address of the master, and **master** is the host name of the master. Clients with this configuration will resynchronize the master if it restarts.

   On the client systems which are not to be direct clients of the master, the `/etc/chrony.conf` file should be the same except that the `local` and `allow` directives should be omitted.
In an isolated network, you can also use the local directive that enables a local reference mode, which allows chronyd operating as an NTP server to appear synchronized to real time, even when it was never synchronized or the last update of the clock happened a long time ago.

To allow multiple servers in the network to use the same local configuration and to be synchronized to one another, without confusing clients that poll more than one server, use the orphan option of the local directive which enables the orphan mode. Each server needs to be configured to poll all other servers with local. This ensures that only the server with the smallest reference ID has the local reference active and other servers are synchronized to it. When the server fails, another one will take over.

28.5. CONFIGURING REMOTE MONITORING ACCESS

chronyc can access chronyd in two ways:

- Internet Protocol, IPv4 or IPv6.
- Unix domain socket, which is accessible locally by the root or chrony user.

By default, chronyc connects to the Unix domain socket. The default path is /var/run/chrony/chronyd.sock. If this connection fails, which can happen for example when chronyc is running under a non-root user, chronyc tries to connect to 127.0.0.1 and then ::1.

Only the following monitoring commands, which do not affect the behavior of chronyd, are allowed from the network:

- activity
- manual list
- rtcdatal
- smoothing
- sources
- sourcestats
- tracking
- waitsync

The set of hosts from which chronyd accepts these commands can be configured with the cmdallow directive in the configuration file of chronyd, or the cmdallow command in chronyc. By default, the commands are accepted only from localhost (127.0.0.1 or ::1).

All other commands are allowed only through the Unix domain socket. When sent over the network, chronyd responds with a Not authorised error, even if it is from localhost.

The following procedure describes how to access chronyd remotely with chronyc.

Procedure

1. Allow access from both IPv4 and IPv6 addresses by adding the following to the /etc/chrony.conf file:
bindcmdaddress 0.0.0.0

or

bindcmdaddress ::

2. Allow commands from the remote IP address, network, or subnet by using the `cmdallow` directive.
Add the following content to the `/etc/chrony.conf` file:

```
  cmdallow 192.168.1.0/24
```

3. Open port 323 in the firewall to connect from a remote system:

```
  # firewall-cmd --zone=public --add-port=323/udp
```

Optionally, you can open port 323 permanently using the `--permanent` option:

```
  # firewall-cmd --permanent --zone=public --add-port=323/udp
```

4. If you opened port 323 permanently, reload the firewall configuration:

```
  firewall-cmd --reload
```

Additional resources

- `chrony.conf(5)` man page

### 28.6. MANAGING TIME SYNCHRONIZATION USING RHEL SYSTEM ROLES

You can manage time synchronization on multiple target machines using the `timesync` role. The `timesync` role installs and configures an NTP or PTP implementation to operate as an NTP client or PTP slave in order to synchronize the system clock with NTP servers or grandmasters in PTP domains.

**WARNING**

The `timesync` role replaces the configuration of the given or detected provider service on the managed host. Previous settings are lost, even if they are not specified in the role variables. The only preserved setting is the choice of provider if the `timesync_ntp_provider` variable is not defined.

The following example shows how to apply the `timesync` role in a situation with just one pool of servers.

**Example 28.1. An example playbook applying the timesync role for a single pool of servers**
- hosts: timesync-test
  vars:
    timesync_ntp_servers:
      - hostname: 2.rhel.pool.ntp.org
        pool: yes
        iburst: yes
    roles:
      - rhel-system-roles.timesync

For a detailed reference on timesync role variables, install the rhel-system-roles package, and see the README.md or README.html files in the /usr/share/doc/rhel-system-roles/timesync directory.

Additional resources

- Preparing a control node and managed nodes to use RHEL System Roles

28.7. ADDITIONAL RESOURCES

- chronyc(1) man page
- chronyd(8) man page
- Frequently Asked Questions
CHAPTER 29. CHRONY WITH HW TIMESTAMPING

Hardware timestamping is a feature supported in some Network Interface Controller (NICs) which provides accurate timestamping of incoming and outgoing packets. NTP timestamps are usually created by the kernel and **chronyd** with the use of the system clock. However, when HW timestamping is enabled, the NIC uses its own clock to generate the timestamps when packets are entering or leaving the link layer or the physical layer. When used with NTP, hardware timestamping can significantly improve the accuracy of synchronization. For best accuracy, both NTP servers and NTP clients need to use hardware timestamping. Under ideal conditions, a sub-microsecond accuracy may be possible.

Another protocol for time synchronization that uses hardware timestamping is **PTP**.

Unlike NTP, PTP relies on assistance in network switches and routers. If you want to reach the best accuracy of synchronization, use PTP on networks that have switches and routers with PTP support, and prefer NTP on networks that do not have such switches and routers.

The following sections describe how to:

- Verify support for hardware timestamping
- Enable hardware timestamping
- Configure client polling interval
- Enable interleaved mode
- Configure server for large number of clients
- Verify hardware timestamping
- Configure PTP-NTP bridge

### 29.1. VERIFYING SUPPORT FOR HARDWARE TIMESTAMPING

To verify that hardware timestamping with NTP is supported by an interface, use the **ethtool -T** command. An interface can be used for hardware timestamping with NTP if ethtool lists the SOF_TIMESTAMPING_TX_HARDWARE and SOF_TIMESTAMPING_TX_SOFTWARE capabilities and also the HWTSTAMP_FILTER_ALL filter mode.

**Example 29.1. Verifying support for hardware timestamping on a specific interface**

```bash
# ethtool -T eth0
```

Output:

```
Timestamping parameters for eth0:
Capabilities:
  hardware-transmit     (SOF_TIMESTAMPING_TX_HARDWARE)
  software-transmit     (SOF_TIMESTAMPING_TX_SOFTWARE)
  hardware-receive      (SOF_TIMESTAMPING_RX_HARDWARE)
  software-receive      (SOF_TIMESTAMPING_RX_SOFTWARE)
  software-system-clock (SOF_TIMESTAMPING_SOFTWARE)
  hardware-raw-clock    (SOF_TIMESTAMPING_RAW_HARDWARE)

PTP Hardware Clock: 0
Hardware Transmit Timestamp Modes:
```

Red Hat Enterprise Linux 9 Configuring basic system settings
CHAPTER 29. CHRONY WITH HW TIMESTAMPING

off (HWTSTAMP_TX_OFF)
on (HWTSTAMP_TX_ON)

Hardware Receive Filter Modes:
none (HWTSTAMP_FILTER_NONE)
all (HWTSTAMP_FILTER_ALL)
ptpv1-l4-sync (HWTSTAMP_FILTER_PTP_V1_L4_SYNC)
ptpv1-l4-delay-req (HWTSTAMP_FILTER_PTP_V1_L4_DELAY_REQ)
ptpv2-l4-sync (HWTSTAMP_FILTER_PTP_V2_L4_SYNC)
ptpv2-l4-delay-req (HWTSTAMP_FILTER_PTP_V2_L4_DELAY_REQ)
ptpv2-l2-sync (HWTSTAMP_FILTER_PTP_V2_L2_SYNC)
ptpv2-l2-delay-req (HWTSTAMP_FILTER_PTP_V2_L2_DELAY_REQ)
ptpv2-event (HWTSTAMP_FILTER_PTP_V2_EVENT)
ptpv2-sync (HWTSTAMP_FILTER_PTP_V2_SYNC)
ptpv2-delay-req (HWTSTAMP_FILTER_PTP_V2_DELAY_REQ)

29.2. ENABLING HARDWARE TIMESTAMPING

To enable hardware timestamping, use the `hwtimestamp` directive in the `/etc/chrony.conf` file. The directive can either specify a single interface, or a wildcard character can be used to enable hardware timestamping on all interfaces that support it. Use the wildcard specification in case that no other application, like `ptp4l` from the `linuxptp` package, is using hardware timestamping on an interface. Multiple `hwtimestamp` directives are allowed in the chrony configuration file.

Example 29.2. Enabling hardware timestamping by using the hwtimestamp directive

```
hwtimestamp eth0
hwtimestamp eth1
hwtimestamp *
```

29.3. CONFIGURING CLIENT POLLING INTERVAL

The default range of a polling interval (64–1024 seconds) is recommended for servers on the Internet. For local servers and hardware timestamping, a shorter polling interval needs to be configured in order to minimize offset of the system clock.

The following directive in `/etc/chrony.conf` specifies a local NTP server using one second polling interval:

```
server ntp.local minpoll 0 maxpoll 0
```

29.4. ENABLING INTERLEAVED MODE

NTP servers that are not hardware NTP appliances, but rather general purpose computers running a software NTP implementation, like `chrony`, will get a hardware transmit timestamp only after sending a packet. This behavior prevents the server from saving the timestamp in the packet to which it corresponds. In order to enable NTP clients receiving transmit timestamps that were generated after the transmission, configure the clients to use the NTP interleaved mode by adding the `xleave` option to the server directive in `/etc/chrony.conf`:

```
server ntp.local minpoll 0 maxpoll 0 xleave
```
29.5. CONFIGURING SERVER FOR LARGE NUMBER OF CLIENTS

The default server configuration allows a few thousands of clients at most to use the interleaved mode concurrently. To configure the server for a larger number of clients, increase the `clientloglimit` directive in `/etc/chrony.conf`. This directive specifies the maximum size of memory allocated for logging of clients' access on the server:

```
clientloglimit 100000000
```

29.6. VERIFYING HARDWARE TIMESTAMPING

To verify that the interface has successfully enabled hardware timestamping, check the system log. The log should contain a message from `chronyd` for each interface with successfully enabled hardware timestamping.

```
Example 29.3. Log messages for interfaces with enabled hardware timestamping

chronyd[4081]: Enabled HW timestamping on eth0
chronyd[4081]: Enabled HW timestamping on eth1
```

When `chronyd` is configured as an NTP client or peer, you can have the transmit and receive timestamping modes and the interleaved mode reported for each NTP source by the `chronyc ntpdata` command:

```
Example 29.4. Reporting the transmit, receive timestamping and interleaved mode for each NTP source

# chronyc ntpdata

Output:

Remote address   : 203.0.113.15 (CB00710F)
Remote port      : 123
Local address    : 203.0.113.74 (CB00714A)
Leap status      : Normal
Version          : 4
Mode             : Server
Stratum          : 1
Poll interval    : 0 (1 seconds)
Precision        : -24 (0.000000060 seconds)
Root delay       : 0.000015 seconds
Root dispersion  : 0.000015 seconds
Reference ID     : 47505300 (GPS)
Reference time   : Wed May 03 13:47:45 2017
Offset           : -0.000000134 seconds
Peer delay       : 0.00005396 seconds
Peer dispersion  : 0.000002329 seconds
Response time    : 0.00152073 seconds
Jitter asymmetry : +0.00
NTP tests        : 111 111 1111
Interleaved      : Yes
Authenticated    : No
TX timestamping : Hardware  
RX timestamping : Hardware  
Total TX : 27  
Total RX : 27  
Total valid RX : 27

Example 29.5. Reporting the stability of NTP measurements

```
# chronyc sourcestats
```

With hardware timestamping enabled, stability of NTP measurements should be in tens or hundreds of nanoseconds, under normal load. This stability is reported in the Std Dev column of the output of the chronyc sourcestats command:

```
Output:
210 Number of sources = 1
Name/IP Address NP NR Span Frequency Freq Skew Offset Std Dev
ntp.local 12 7 11 +0.000 0.019 +0ns 49ns
```

29.7. CONFIGURING PTP-NTP BRIDGE

If a highly accurate Precision Time Protocol (PTP) grandmaster is available in a network that does not have switches or routers with PTP support, a computer may be dedicated to operate as a PTP slave and a stratum-1 NTP server. Such a computer needs to have two or more network interfaces, and be close to the grandmaster or have a direct connection to it. This will ensure highly accurate synchronization in the network.

Configure the ptp4l and phc2sys programs from the linuxptp packages to use one interface to synchronize the system clock using PTP.

Configure chronyd to provide the system time using the other interface:

```
Example 29.6. Configuring chronyd to provide the system time using the other interface
bindaddress 203.0.113.74  
hwtimestamp eth1  
local stratum 1
```
CHAPTER 30. OVERVIEW OF NETWORK TIME SECURITY (NTS) IN CHRONY

Network Time Security (NTS) is an authentication mechanism for Network Time Protocol (NTP), designed to scale substantial clients. It verifies that the packets received from the server machines are unaltered while moving to the client machine. Network Time Security (NTS) includes a Key Establishment (NTS-KE) protocol that automatically creates the encryption keys used between the server and its clients.

30.1. ENABLING NETWORK TIME SECURITY (NTS) IN THE CLIENT CONFIGURATION FILE

By default, Network Time Security (NTS) is not enabled. You can enable NTS in the /etc/chrony.conf.

For that, perform the following steps:

Prerequisites

- Server with the NTS support

Procedure

In the client configuration file:

1. Specify the server with the nts option in addition to the recommended iburst option.
   For example:
   ```
   server time.example.com iburst nts
   server nts.netnod.se iburst nts
   server ptbtime1.ptb.de iburst nts
   ```

2. To avoid repeating the Network Time Security-Key Establishment (NTS-KE) session during system boot, add the following line to chrony.conf, if it is not present:
   ```
   ntsdumpdir /var/lib/chrony
   ```

3. To disable synchronization with Network Time Protocol (NTP) servers provided by DHCP, comment out or remove the following line in chrony.conf, if it is present:
   ```
   sourcedir /run/chrony-dhcp
   ```

4. Save your changes.

5. Restart the chronyd service:
   ```
   systemctl restart chronyd
   ```

Verification

- Verify if the NTS keys were successfully established:
  ```
  # chronyc -N authdata
  ```
### Additional resources

- [chrony.conf(5) man page](#)

### 30.2. ENABLING NETWORK TIME SECURITY (NTS) ON THE SERVER

If you run your own Network Time Protocol (NTP) server, you can enable the server Network Time Security (NTS) support to facilitate its clients to synchronize securely.

If the NTP server is a client of other servers, that is, it is not a Stratum 1 server, it should use NTS or symmetric key for its synchronization.

#### Prerequisites

- Server private key in **PEM** format
- Server certificate with required intermediate certificates in **PEM** format

#### Procedure

1. Specify the private key and the certificate file in `chrony.conf`

   For example:
   ```
   ntsseverkey /etc/pki/tls/private/foo.example.net.key
   ntssevercert /etc/pki/tls/certs/foo.example.net.crt
   ```

2. Ensure that both the key and certificate files are readable by the chrony system user, by setting the group ownership.

   For example:
   ```
   chown :chrony /etc/pki/tls/**/foo.example.net.*
   ```
3. Ensure the `ntsdumpdir /var/lib/chrony` directive is present in the `chrony.conf`.

4. Restart the `chronyd` service:

   ```
   systemctl restart chronyd
   ```

   **IMPORTANT**

   If the server has a firewall, it needs to allow both the **UDP 123** and **TCP 4460** ports for NTP and Network Time Security-Key Establishment (NTS-KE).

**Verification**

- Perform a quick test from a client machine with the following command:

  ```
  $ chronyd -Q -t 3 'server
  foo.example.net iburst nts maxsamples 1'
  ```

  2021-09-15T13:45:26Z chronyd version 4.1 starting (+CMDMON +NTP +REFCLOCK +RTC +PRIVDROP +SCFILTER +SIGND +ASYNCDNS +NTS +SECHASH +IPV6 +DEBUG)
  2021-09-15T13:45:26Z Disabled control of system clock
  2021-09-15T13:45:28Z System clock wrong by 0.002205 seconds (ignored)
  2021-09-15T13:45:28Z chronyd exiting

  The **System clock wrong** message indicates the NTP server is accepting NTS-KE connections and responding with NTS-protected NTP messages.

- Verify the NTS-KE connections and authenticated NTP packets observed on the server:

  ```
  # chronyc serverstats
  ```

  NTP packets received : 7
  NTP packets dropped : 0
  Command packets received : 22
  Command packets dropped : 0
  Client log records dropped : 0
  NTS-KE connections accepted: 1
  NTS-KE connections dropped : 0
  Authenticated NTP packets: 7

  If the value of the **NTS-KE connections accepted** and **Authenticated NTP packets** field is a non-zero value, it means that at least one client was able to connect to the NTS-KE port and send an authenticated NTP request.
CHAPTER 31. USING SECURE COMMUNICATIONS BETWEEN TWO SYSTEMS WITH OPENSSH

SSH (Secure Shell) is a protocol which provides secure communications between two systems using a client-server architecture and allows users to log in to server host systems remotely. Unlike other remote communication protocols, such as FTP or Telnet, SSH encrypts the login session, which prevents intruders to collect unencrypted passwords from the connection.

Red Hat Enterprise Linux includes the basic OpenSSH packages: the general openssh package, the openssh-server package and the openssh-clients package. Note that the OpenSSH packages require the OpenSSL package openssl-libs, which installs several important cryptographic libraries that enable OpenSSH to provide encrypted communications.

31.1. SSH AND OPENSSH

SSH (Secure Shell) is a program for logging into a remote machine and executing commands on that machine. The SSH protocol provides secure encrypted communications between two untrusted hosts over an insecure network. You can also forward X11 connections and arbitrary TCP/IP ports over the secure channel.

The SSH protocol mitigates security threats, such as interception of communication between two systems and impersonation of a particular host, when you use it for remote shell login or file copying. This is because the SSH client and server use digital signatures to verify their identities. Additionally, all communication between the client and server systems is encrypted.

A host key authenticates hosts in the SSH protocol. Host keys are cryptographic keys that are generated automatically when OpenSSH is first installed, or when the host boots for the first time.

OpenSSH is an implementation of the SSH protocol supported by Linux, UNIX, and similar operating systems. It includes the core files necessary for both the OpenSSH client and server. The OpenSSH suite consists of the following user-space tools:

- **ssh** is a remote login program (SSH client).
- **sshd** is an OpenSSH SSH daemon.
- **scp** is a secure remote file copy program.
- **sftp** is a secure file transfer program.
- **ssh-agent** is an authentication agent for caching private keys.
- **ssh-add** adds private key identities to **ssh-agent**.
- **ssh-keygen** generates, manages, and converts authentication keys for **ssh**.
- **ssh-copy-id** is a script that adds local public keys to the **authorized_keys** file on a remote SSH server.
- **ssh-keyscan** gathers SSH public host keys.
NOTE

In RHEL 9, the Secure copy protocol (SCP) is replaced with the SSH File Transfer Protocol (SFTP) by default. This is because SCP has already caused security issues, for example CVE-2020-15778.

If SFTP is unavailable or incompatible in your scenario, you can use the -O option to force use of the original SCP/RCP protocol.

For additional information, see the OpenSSH SCP protocol deprecation in Red Hat Enterprise Linux 9 article.

Two versions of SSH currently exist: version 1, and the newer version 2. The OpenSSH suite in RHEL supports only SSH version 2. It has an enhanced key-exchange algorithm that is not vulnerable to exploits known in version 1.

OpenSSH, as one of core cryptographic subsystems of RHEL, uses system-wide crypto policies. This ensures that weak cipher suites and cryptographic algorithms are disabled in the default configuration. To modify the policy, the administrator must either use the update-crypto-policies command to adjust the settings or manually opt out of the system-wide crypto policies.

The OpenSSH suite uses two sets of configuration files: one for client programs (that is, ssh, scp, and sftp), and another for the server (the sshd daemon).

System-wide SSH configuration information is stored in the /etc/ssh/ directory. User-specific SSH configuration information is stored in ~/.ssh/ in the user's home directory. For a detailed list of OpenSSH configuration files, see the FILES section in the sshd(8) man page.

Additional resources

- Man pages listed by using the man -k ssh command
- Using system-wide cryptographic policies

31.2. CONFIGURING AND STARTING AN OPENSSH SERVER

Use the following procedure for a basic configuration that might be required for your environment and for starting an OpenSSH server. Note that after the default RHEL installation, the sshd daemon is already started and server host keys are automatically created.

Prerequisites

- The openssh-server package is installed.

Procedure

1. Start the sshd daemon in the current session and set it to start automatically at boot time:

```bash
# systemctl start sshd
# systemctl enable sshd
```

2. To specify different addresses than the default 0.0.0.0 (IPv4) or :: (IPv6) for the ListenAddress directive in the /etc/ssh/sshd_config configuration file and to use a slower dynamic network configuration, add the dependency on the network-online.target target unit
to the `sshd.service` unit file. To achieve this, create the
`/etc/systemd/system/sshd.service.d/local.conf` file with the following content:

```
[Unit]
Wants=network-online.target
After=network-online.target
```

3. Review if OpenSSH server settings in the `/etc/ssh/sshd_config` configuration file meet the requirements of your scenario.

4. Optionally, change the welcome message that your OpenSSH server displays before a client authenticates by editing the `/etc/issue` file, for example:

```
Welcome to ssh-server.example.com
Warning: By accessing this server, you agree to the referenced terms and conditions.
```

Ensure that the `Banner` option is not commented out in `/etc/ssh/sshd_config` and its value contains `/etc/issue`:

```
# less /etc/ssh/sshd_config | grep Banner
Banner /etc/issue
```

Note that to change the message displayed after a successful login you have to edit the `/etc/motd` file on the server. See the `pam_motd` man page for more information.

5. Reload the `systemd` configuration and restart `sshd` to apply the changes:

```
# systemctl daemon-reload
# systemctl restart sshd
```

**Verification**

1. Check that the `sshd` daemon is running:

```
# systemctl status sshd
● sshd.service - OpenSSH server daemon
 Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled; vendor preset: enabled)
 Active: active (running) since Mon 2019-11-18 14:59:58 CET; 6min ago
 Docs: man:sshd(8)
         man:sshd_config(5)
 Main PID: 1149 (sshd)
 Tasks: 1 (limit: 11491)
 Memory: 1.9M
 CGroup: /system.slice/sshd.service
         └─1149 /usr/sbin/sshd -D -oCiphers=aes128-ctr,aes256-ctr,aes128-cbc,aes256-cbc -oMACs=hmac-sha2-256,>

Nov 18 14:59:58 ssh-server-example.com systemd[1]: Starting OpenSSH server daemon...
Nov 18 14:59:58 ssh-server-example.com sshd[1149]: Server listening on 0.0.0.0 port 22.
Nov 18 14:59:58 ssh-server-example.com sshd[1149]: Server listening on :: port 22.
Nov 18 14:59:58 ssh-server-example.com systemd[1]: Started OpenSSH server daemon.
```

2. Connect to the SSH server with an SSH client.
# ssh user@ssh-server-example.com
ECDSA key fingerprint is SHA256:dXbaS0RG/UzITTku8GtXSz0S1++IPegSy31v3L/FAEc.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ssh-server-example.com' (ECDSA) to the list of known hosts.

user@ssh-server-example.com's password:

Additional resources

- `sshd(8)` and `sshd_config(5)` man pages.

### 31.3. SETTING AN OPENSSH SERVER FOR KEY-BASED AUTHENTICATION

To improve system security, enforce key-based authentication by disabling password authentication on your OpenSSH server.

**Prerequisites**

- The `openssh-server` package is installed.
- The `sshd` daemon is running on the server.

**Procedure**

1. Open the `/etc/ssh/sshd_config` configuration in a text editor, for example:

   ```
   # vi /etc/ssh/sshd_config
   ```

2. Change the `PasswordAuthentication` option to `no`:

   ```
   PasswordAuthentication no
   ```

   On a system other than a new default installation, check that `PubkeyAuthentication no` has not been set and the `ChallengeResponseAuthentication` directive is set to `no`. If you are connected remotely, not using console or out-of-band access, test the key-based login process before disabling password authentication.

3. To use key-based authentication with NFS-mounted home directories, enable the `use_nfs_home_dirs` SELinux boolean:

   ```
   # setsebool -P use_nfs_home_dirs 1
   ```

4. Reload the `sshd` daemon to apply the changes:

   ```
   # systemctl reload sshd
   ```

**Additional resources**

- `sshd(8)`, `sshd_config(5)`, and `setsebool(8)` man pages.
31.4. GENERATING SSH KEY PAIRS

Use this procedure to generate an SSH key pair on a local system and to copy the generated public key to an OpenSSH server. If the server is configured accordingly, you can log in to the OpenSSH server without providing any password.

IMPORTANT

If you complete the following steps as root, only root is able to use the keys.

Procedure

1. To generate an ECDSA key pair for version 2 of the SSH protocol:

   $ ssh-keygen -t ecdsa
   Generating public/private ecdsa key pair.
   Enter file in which to save the key (/home/joesec/.ssh/id_ecdsa):
   Enter passphrase (empty for no passphrase):
   Enter same passphrase again:
   Your identification has been saved in /home/joesec/.ssh/id_ecdsa.
   Your public key has been saved in /home/joesec/.ssh/id_ecdsa.pub.
   The key fingerprint is:
   SHA256:Q/x+qms4j7PCQ0qFd09iZEFHA+SqwBKRNaU72oZfaCl
   joesec@localhost.example.com
   The key's randomart image is:
   +---[ECDSA 256]---+
   |.oo..o=++        |
   |.. o .oo .       |
   |. .. o. o        |
   |....o.+...       |
   |0.00.0 +S .      |
   |.=.+.   .o       |
   |E.*+.  .  . .    |
   |.=..+ +..  o     |
   |  .  oo*+o.      |
   +----[SHA256]-----+

   You can also generate an RSA key pair by using the -t rsa option with the ssh-keygen command or an Ed25519 key pair by entering the ssh-keygen -t ed25519 command.

2. To copy the public key to a remote machine:

   $ ssh-copy-id joesec@ssh-server-example.com
   /usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
   joesec@ssh-server-example.com's password:
   ...
   Number of key(s) added: 1

   Now try logging into the machine, with: "ssh 'joesec@ssh-server-example.com'" and check to make sure that only the key(s) you wanted were added.

If you do not use the ssh-agent program in your session, the previous command copies the most recently modified ~/.ssh/id*.pub public key if it is not yet installed. To specify another public-key file or to prioritize keys in files over keys cached in memory by ssh-agent, use the
**NOTE**

If you reinstall your system and want to keep previously generated key pairs, back up the `~/.ssh/` directory. After reinstalling, copy it back to your home directory. You can do this for all users on your system, including **root**.

**Verification**

1. Log in to the OpenSSH server without providing any password:

   ```sh
   $ ssh joesec@ssh-server-example.com
   Welcome message.
   ...
   Last login: Mon Nov 18 18:28:42 2019 from ::1
   ```

**Additional resources**

- `ssh-keygen(1)` and `ssh-copy-id(1)` man pages.

### 31.5. USING SSH KEYS STORED ON A SMART CARD

Red Hat Enterprise Linux enables you to use RSA and ECDSA keys stored on a smart card on OpenSSH clients. Use this procedure to enable authentication using a smart card instead of using a password.

**Prerequisites**

- On the client side, the `opensc` package is installed and the `pcscd` service is running.

**Procedure**

1. List all keys provided by the OpenSC PKCS #11 module including their PKCS #11 URIs and save the output to the `keys.pub` file:

   ```sh
   $ ssh-keygen -D pkcs11: > keys.pub
   $ ssh-keygen -D pkcs11:
   ssh-rsa AAAAB3NzaC1yc2E...KKZMzcQZxz
   pkcs11:id=%02;object=SIGN%20pubkey;token=SSH%20key;manufacturer=piv_II?module-path=/usr/lib64/pkcs11/opensc-pkcs11.so
   ecdsa-sha2-nistp256 AAA...J0hkYnnsM=
   pkcs11:id=%01;object=PIV%20AUTH%20pubkey;token=SSH%20key;manufacturer=piv_II?
   module-path=/usr/lib64/pkcs11/opensc-pkcs11.so
   ```

2. To enable authentication using a smart card on a remote server (`example.com`), transfer the public key to the remote server. Use the `ssh-copy-id` command with `keys.pub` created in the previous step:

   ```sh
   $ ssh-copy-id -f -i keys.pub username@example.com
   ```

3. To connect to `example.com` using the ECDSA key from the output of the `ssh-keygen -D` command in step 1, you can use just a subset of the URI, which uniquely references your key, for example:
4. You can use the same URI string in the ~/.ssh/config file to make the configuration permanent:

```
$ cat ~/.ssh/config
IdentityFile "pkcs11:id=%01?module-path=/usr/lib64/pkcs11/opensc-pkcs11.so"
$ ssh example.com
Enter PIN for 'SSH key':
[example.com] $
```

Because OpenSSH uses the p11-kit-proxy wrapper and the OpenSC PKCS #11 module is registered to PKCS#11 Kit, you can simplify the previous commands:

```
$ ssh -i "pkcs11:id=%01" example.com
Enter PIN for 'SSH key':
[example.com] $
```

If you skip the id= part of a PKCS #11 URI, OpenSSH loads all keys that are available in the proxy module. This can reduce the amount of typing required:

```
$ ssh -i pkcs11: example.com
Enter PIN for 'SSH key':
[example.com] $
```

Additional resources

- Fedora 28: Better smart card support in OpenSSH
- p11-kit(8), opensc.conf(5), pcscd(8), ssh(1), and ssh-keygen(1) man pages

### 31.6. MAKING OPENSSH MORE SECURE

The following tips help you to increase security when using OpenSSH. Note that changes in the /etc/ssh/sshd_config OpenSSH configuration file require reloading the sshd daemon to take effect:

```
# systemctl reload sshd
```

**IMPORTANT**

The majority of security hardening configuration changes reduce compatibility with clients that do not support up-to-date algorithms or cipher suites.

**Disabling insecure connection protocols**

- To make SSH truly effective, prevent the use of insecure connection protocols that are replaced by the OpenSSH suite. Otherwise, a user’s password might be protected using SSH for one session only to be captured later when logging in using Telnet. For this reason, consider disabling insecure protocols, such as telnet, rsh, rlogin, and ftp.

**Enabling key-based authentication and disabling password-based authentication**
Disabling passwords for authentication and allowing only key pairs reduces the attack surface and it also might save users’ time. On clients, generate key pairs using the `ssh-keygen` tool and use the `ssh-copy-id` utility to copy public keys from clients on the OpenSSH server. To disable password-based authentication on your OpenSSH server, edit `/etc/ssh/sshd_config` and change the `PasswordAuthentication` option to `no`:

```
PasswordAuthentication no
```

Key types

- Although the `ssh-keygen` command generates a pair of RSA keys by default, you can instruct it to generate ECDSA or Ed25519 keys by using the `-t` option. The ECDSA (Elliptic Curve Digital Signature Algorithm) offers better performance than RSA at the equivalent symmetric key strength. It also generates shorter keys. The Ed25519 public-key algorithm is an implementation of twisted Edwards curves that is more secure and also faster than RSA, DSA, and ECDSA. OpenSSH creates RSA, ECDSA, and Ed25519 server host keys automatically if they are missing. To configure the host key creation in RHEL, use the `sshd-keygen@.service` instantiated service. For example, to disable the automatic creation of the RSA key type:

```
# systemctl mask sshd-keygen@rsa.service
```

- To exclude particular key types for SSH connections, comment out the relevant lines in `/etc/ssh/sshd_config`, and reload the `sshd` service. For example, to allow only Ed25519 host keys:

```
# HostKey /etc/ssh/ssh_host_rsa_key
# HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key
```

Non-default port

- By default, the `sshd` daemon listens on TCP port 22. Changing the port reduces the exposure of the system to attacks based on automated network scanning and thus increase security through obscurity. You can specify the port using the `Port` directive in the `/etc/ssh/sshd_config` configuration file. You also have to update the default SELinux policy to allow the use of a non-default port. To do so, use the `semanage` tool from the `policycoreutils-python-utils` package:

```
# semanage port -a -t ssh_port_t -p tcp port_number
```

Furthermore, update `firewalld` configuration:

```
# firewall-cmd --add-port port_number/tcp
# firewall-cmd --runtime-to-permanent
```

In the previous commands, replace `port_number` with the new port number specified using the `Port` directive.

Root login

- `PermitRootLogin` is set to `prohibit-password` by default. This enforces the use of key-based authentication instead of the use of passwords for logging in as root and reduces risks by preventing brute-force attacks.
CAUTION

Enabling logging in as the root user is not a secure practice because the administrator cannot audit which users run which privileged commands. For using administrative commands, log in and use `sudo` instead.

Using the X Security extension

- The X server in Red Hat Enterprise Linux clients does not provide the X Security extension. Therefore, clients cannot request another security layer when connecting to untrusted SSH servers with X11 forwarding. Most applications are not able to run with this extension enabled anyway.

  By default, the `ForwardX11Trusted` option in the `/etc/ssh/ssh_config.d/05-redhat.conf` file is set to `yes`, and there is no difference between the `ssh -X remote_machine` (untrusted host) and `ssh -Y remote_machine` (trusted host) command.

If your scenario does not require the X11 forwarding feature at all, set the `X11Forwarding` directive in the `/etc/ssh/sshd_config` configuration file to `no`.

Restricting access to specific users, groups, or domains

- The `AllowUsers` and `AllowGroups` directives in the `/etc/ssh/sshd_config` configuration file server enable you to permit only certain users, domains, or groups to connect to your OpenSSH server. You can combine `AllowUsers` and `AllowGroups` to restrict access more precisely, for example:

  ```
  AllowUsers *@192.168.1.*, *@10.0.0.*, !*@192.168.1.2
  AllowGroups example-group
  ```

  The previous configuration lines accept connections from all users from systems in 192.168.1.* and 10.0.0.* subnets except from the system with the 192.168.1.2 address. All users must be in the `example-group` group. The OpenSSH server denies all other connections.

  Note that using allowlists (directives starting with Allow) is more secure than using blocklists (options starting with Deny) because allowlists block also new unauthorized users or groups.

Changing system-wide cryptographic policies

- OpenSSH uses RHEL system-wide cryptographic policies, and the default system-wide cryptographic policy level offers secure settings for current threat models. To make your cryptographic settings more strict, change the current policy level:

  ```
  # update-crypto-policies --set FUTURE
  Setting system policy to FUTURE
  ```

- To opt-out of the system-wide crypto policies for your OpenSSH server, uncomment the line with the `CRYPTO_POLICY=` variable in the `/etc/sysconfig/sshd` file. After this change, values that you specify in the `Ciphers`, `MACs`, `KexAlgorithms`, and `GSSAPIKexAlgorithms` sections in the `/etc/ssh/sshd_config` file are not overridden. Note that this task requires deep expertise in configuring cryptographic options.

- See `Using system-wide cryptographic policies` in the `Security hardening` title for more information.

Additional resources
31.7. CONNECTING TO A REMOTE SERVER USING AN SSH JUMP HOST

Use this procedure for connecting your local system to a remote server through an intermediary server, also called jump host.

Prerequisites

- A jump host accepts SSH connections from your local system.
- A remote server accepts SSH connections only from the jump host.

Procedure

1. Define the jump host by editing the ~/.ssh/config file on your local system, for example:

   ```
   Host jump-server1
   HostName jump1.example.com
   ```

   - The **Host** parameter defines a name or alias for the host you can use in ssh commands. The value can match the real host name, but can also be any string.
   - The **HostName** parameter sets the actual host name or IP address of the jump host.

2. Add the remote server jump configuration with the **ProxyJump** directive to ~/.ssh/config file on your local system, for example:

   ```
   Host remote-server
   HostName remote1.example.com
   ProxyJump jump-server1
   ```

3. Use your local system to connect to the remote server through the jump server:

   ```
   $ ssh remote-server
   ```

   The previous command is equivalent to the ssh -J jump-server1 remote-server command if you omit the configuration steps 1 and 2.
NOTE

You can specify more jump servers and you can also skip adding host definitions to the configurations file when you provide their complete host names, for example:

```
$ ssh -J jump1.example.com,jump2.example.com,jump3.example.com remote1.example.com
```

Change the host name-only notation in the previous command if the user names or SSH ports on the jump servers differ from the names and ports on the remote server, for example:

```
$ ssh -J johndoe@jump1.example.com:75,johndoe@jump2.example.com:75,johndoe@jump3.example.com:75 joesec@remote1.example.com:220
```

Additional resources

- `ssh_config(5)` and `ssh(1)` man pages.

### 31.8. CONNECTING TO REMOTE MACHINES WITH SSH KEYS USING SSH-AGENT

To avoid entering a passphrase each time you initiate an SSH connection, you can use the `ssh-agent` utility to cache the private SSH key. The private key and the passphrase remain secure.

**Prerequisites**

- You have a remote host with SSH daemon running and reachable through the network.
- You know the IP address or hostname and credentials to log in to the remote host.
- You have generated an SSH key pair with a passphrase and transferred the public key to the remote machine.

For more information, see [Generating SSH key pairs](#).

**Procedure**

1. Optional: Verify you can use the key to authenticate to the remote host:
   
   a. Connect to the remote host using SSH:

   ```
   $ ssh example.user1@198.51.100.1 hostname
   ```

   b. Enter the passphrase you set while creating the key to grant access to the private key.

   ```
   $ ssh example.user1@198.51.100.1 hostname
   ```

2. Start the `ssh-agent`.

---

CHAPTER 31. USING SECURE COMMUNICATIONS BETWEEN TWO SYSTEMS WITH OPENSSH
$ eval $(ssh-agent)
Agent pid 20062

3. Add the key to ssh-agent.

$ ssh-add ~/.ssh/id_rsa
Enter passphrase for ~/.ssh/id_rsa:
Identity added: ~/.ssh/id_rsa (example.user0@198.51.100.12)

Verification

- Optional: Log in to the host machine using SSH.

$ ssh example.user1@198.51.100.1

Last login: Mon Sep 14 12:56:37 2020

Note that you did not have to enter the passphrase.

31.9. ADDITIONAL RESOURCES

- sshd(8), ssh(1), scp(1), sftp(1), ssh-keygen(1), ssh-copy-id(1), ssh_config(5), sshd_config(5), update-crypto-policies(8), and crypto-policies(7) man pages.
- OpenSSH Home Page
- Configuring SELinux for applications and services with non-standard configurations
- Controlling network traffic using firewalld