Red Hat Enterprise Linux 9

基本的なシステム設定

システムに必要な機能の設定とシステム環境のカスタマイズ

Last Updated: 2024-06-26

Red Hat Enterprise Linux 9 基本的なシステム設定

システムに必要な機能の設定とシステム環境のカスタマイズ

法律上の通知

概要

基本的なシステム管理タスク、環境設定、システムの登録、ネットワークアクセスとシステムセキュリティーの設定を行います。ユーザー、グループ、ファイルのパーミッションを管理します。複数の RHEL システム上のシステム設定インターフェイスを管理するには、システムロールを使用します。効率的なサービス管理には systemd を使用します。chrony を使用して Network Time Protocol (NTP) を設定します。ReaR を使用してシステムをバックアップおよび復元します。

目次

RED HAT ドキュメントへのフィードバック (英語のみ) 4

第1章 基本的なネットワークアクセスの設定と管理 5
1.1. グラフィカルインストールモードでのネットワークおよびホスト名の設定 5
1.2. NMCLI を使用したイーサネット接続の設定 6
1.3. NMTUIを使用したイーサネット接続の設定 9
1.4. RHEL WEB コンソールにおけるネットワークの管理 12
1.5. RHEL システムロールを使用したネットワークの管理 13
1.6. 関連情報 14

第2章 システム登録およびサブスクリプション管理 15
2.1. インストール後のシステムの登録 15
2.2. WEB コンソールで認証情報を使用してサブスクリプションを登録 16
2.3. GNOME での RED HAT アカウントを使用したシステム登録 17
2.4. GNOME でのアクティベーションキーを使用したシステム登録 18

第3章 RED HAT サポートへのアクセス 21
3.1. RED HAT カスタマーポータルで利用できる RED HAT サポート 21
3.2. SOSREPORT を使用した問題のトラブルシューティング 21

第4章 基本的な環境設定の変更 23
4.1. 日付および時刻の設定 23
4.2. システムロケールの設定 23
4.3. キーボードレイアウトの設定 24
4.4. テキストコンソールモードでフォントサイズの変更 25
4.5. 関連情報 26

第5章 2 台のシステム間で OPENSSH を使用した安全な通信の使用 27
5.1. SSH と OPENSSH 27
5.2. OPENSSH サーバーの設定および起動 28
5.3. 鍵ベースの認証用の OPENSSH サーバーの設定 30
5.4. SSH 鍵ペアの生成 30
5.5. スマートカードに保存された SSH 鍵の使用 32
5.6. OPENSSH のセキュリティーの強化 33
5.7. SSH ジャンプホストを使用してリモートサーバーに接続 37
5.8. SSH-AGENT を使用して SSH キーでリモートマシンに接続する手順 38
5.9. 関連情報 39

第6章 基本的なシステムセキュリティーの設定 40
6.1. ファイアウォールサービスの有効化 40
6.2. 基本的な SELINUX 設定の管理 41
6.3. SELINUX で必要なステータスの確認 41
6.4. 関連情報 42

第7章 RHEL システムロールの概要 43

第8章 ログファイルを使用した問題のトラブルシューティング 46
8.1. SYSLOG メッセージを処理するサービス 46
8.2. SYSLOG メッセージを保存するサブディレクトリー 46
8.3. WEB コンソールでログファイルの検査 46
8.4. コマンドラインでのログの表示 47
8.5. 関連情報 48

第9章 ユーザーおよびグループの管理 49

9.1. ユーザーアカウントおよびグループアカウントの管理の概要 49
9.2. ユーザーアカウントの管理 50
9.3. コマンドラインからのユーザーの管理 53
9.4. WEB コンソールでユーザーアカウントの管理 57
9.5. コマンドラインを使用したユーザーグループの編集 60
9.6. ROOT パスワードの変更およびリセット 64

第10章 SUDO アクセスの管理 68
10.1. SUDOERS のユーザー認可 68
10.2. ユーザーへの SUDO アクセス権限の付与 69
10.3. 非特権ユーザーが特定のコマンドを実行できるようにする 70

第11章 ファイルシステムの権限の管理 73
11.1. ファイル権限の管理 73
11.2. アクセス制御リストの管理 79
11.3. UMASK の管理 81

第12章 SYSTEMD の管理 86
12.1. SYSTEMD のユニットファイルの場所 86
12.2. SYSTEMCTL によるシステムサービス管理 87
12.3. ターゲットシステム状態でのブート 94
12.4. システムのシャットダウン、サスペンド、およびハイバネート 98

第13章 時刻同期の設定 104
13.1. CHRONY スイートを使用した NTP の設定 104
13.2. CHRONY の使用 105
13.3. ハードウェアのタイムスタンプを使用した CHRONY 111
13.4. CHRONY における NETWORK TIME SECURITY (NTS) の概要 115

第14章 システムの復旧および復元 119
14.1. REAR の設定 119
14.2. 64 ビット IBM Z アーキテクチャーで REAR レスキューイメージの使用 120

RED HAT ドキュメントへのフィードバック (英語のみ)

Red Hat ドキュメントに関するご意見やご感想をお寄せください。また、改善点があればお知らせください。

Jira からのフィードバック送信 (アカウントが必要)

1. Jira の Web サイトにログインします。
2. 上部のナビゲーションバーで **Create** をクリックします。
3. **Summary** フィールドにわかりやすいタイトルを入力します。
4. **Description** フィールドに、ドキュメントの改善に関するご意見を記入してください。ドキュメントの該当部分へのリンクも追加してください。
5. ダイアログの下部にある **Create** をクリックします。

第1章 基本的なネットワークアクセスの設定と管理

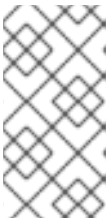
このセクションでは、Red Hat Enterprise Linux でネットワーク設定を設定する方法に関する基本的なオプションについてのみ説明します。

1.1. グラフィカルインストールモードでのネットワークおよびホスト名の設定

以下の手順に従って、ネットワークとホスト名を設定します。

手順

1. **インストール概要** 画面から、**ネットワークとホスト名** をクリックします。
2. 左側のペインのリストから、インターフェイスを選択します。詳細が右側のペインに表示されます。



注記

em1 や **wl3sp0** といった一貫性のある名前をネットワークデバイスの特定に使用するネットワークデバイス命名の標準仕様には、いくつかのタイプがあります。このような標準仕様の詳細は Configuring and managing networking を参照してください。

3. 選択したインタフェースを有効または無効にするには、**ON/OFF** スイッチを切り替えます。

注記

インストールプログラムは、ローカルでアクセス可能なインターフェイスを自動的に検出し、手動で追加または削除できません。

4. **+** をクリックして、仮想ネットワークインターフェイスを追加します。仮想ネットワークインターフェイスは、Team (非推奨)、Bond、Bridge、または VLAN です。
5. **-** を選択して、仮想インターフェイスを削除します。
6. **設定** をクリックして、既存のインターフェイスの IP アドレス、DNS サーバー、またはルーティング設定 (仮想と物理の両方) などの設定を変更します。
7. **ホスト名** フィールドに、システムのホスト名を入力します。

注記

- ホスト名は、**hostname.domainname** 形式の完全修飾ドメイン名 (FQDN)、またはドメインなしの短縮ホスト名のいずれかにします。多くのネットワークには、自動的に接続したシステムにドメイン名を提供する DHCP (Dynamic Host Configuration Protocol) サービスがあります。DHCP サービスがこのシステムにドメイン名を割り当てるようにするには、短縮ホスト名のみを指定します。

- 静的 IP およびホスト名の設定を使用する場合、短縮名または FQDN を使用するかどうかは、計画したシステムのユースケースによって異なります。Red Hat Identity Management はプロビジョニング時に FQDN を設定しますが、サードパーティーのソフトウェア製品によっては短縮名が必要になる場合があります。いずれの場合も、すべての状況で両方のフォームの可用性を確保するには、**IP FQDN short-alias** の形式で **/etc/hosts** にホストのエントリーを追加します。

- **localhost** の値は、ターゲットシステムの静的ホスト名が指定されておらず、(たとえば、DHCP または DNS を使用する NetworkManager による) ネットワーク設定時に、インストールされるシステムの実際のホスト名が設定されることを示しています。

- ホスト名に使用できるのは、英数字と **-** または **.** のみです。ホスト名は 64 文字以下である必要があります。ホスト名は、**-** および **.** で開始したり終了したりできません。DNS に準拠するには、FQDN の各部分は 63 文字以下で、ドットを含む FQDN の合計の長さは 255 文字を超えることができません。

8. **Apply** をクリックして、ホスト名をインストーラー環境に適用します。

9. また、**ネットワークおよびホスト名** 画面では、ワイヤレスオプションを選択できます。右側のペインで **ネットワークの選択** をクリックして Wifi 接続を選択します。必要に応じてパスワードを入力し、**完了** をクリックします。

関連情報

- 高度な RHEL 9 インストールの実行

1.2. NMCLI を使用したイーサネット接続の設定

イーサネット経由でホストをネットワークに接続する場合は、**nmcli** ユーティリティーを使用してコマンドラインで接続の設定を管理できます。

前提条件

- 物理または仮想イーサネットネットワークインターフェイスコントローラー (NIC) がサーバーに設定されている。

手順

1. NetworkManager 接続プロファイルをリストします。

```
# nmcli connection show
NAME                UUID                                  TYPE      DEVICE
Wired connection 1  a5eb6490-cc20-3668-81f8-0314a27f3f75  ethernet  enp1s0
```

デフォルトでは、NetworkManager はホスト内の各 NIC のプロファイルを作成します。この NIC を特定のネットワークにのみ接続する予定がある場合は、自動作成されたプロファイルを調整してください。この NIC をさまざまな設定のネットワークに接続する予定がある場合は、ネットワークごとに個別のプロファイルを作成してください。

2. 追加の接続プロファイルを作成する場合は、次のように入力します。

```
# nmcli connection add con-name <connection-name> ifname <device-name> type ethernet
```

既存のプロファイルを変更するには、この手順をスキップしてください。

3. オプション: 接続プロファイルの名前を変更します。

```
# nmcli connection modify "Wired connection 1" connection.id "Internal-LAN"
```

ホストに複数のプロファイルがある場合は、わかりやすい名前を付けると、プロファイルの目的を識別しやすくなります。

4. 接続プロファイルの現在の設定を表示します。

```
# nmcli connection show Internal-LAN
...
connection.interface-name:     enp1s0
connection.autoconnect:        yes
ipv4.method:                   auto
ipv6.method:                   auto
...
```

5. IPv4 を設定します。

- DHCP を使用するには、次のように入力します。

```
# nmcli connection modify Internal-LAN ipv4.method auto
```

ipv4.method がすでに **auto** (デフォルト) に設定されている場合は、この手順をスキップしてください。

- 静的 IPv4 アドレス、ネットワークマスク、デフォルトゲートウェイ、DNS サーバー、および検索ドメインを設定するには、次のように入力します。

```
# nmcli connection modify Internal-LAN ipv4.method manual ipv4.addresses 192.0.2.1/24 ipv4.gateway 192.0.2.254 ipv4.dns 192.0.2.200 ipv4.dns-search example.com
```

6. IPv6 設定を行います。

- ステートレスアドレス自動設定 (SLAAC) を使用するには、次のように入力します。

```
# nmcli connection modify Internal-LAN ipv6.method auto
```

ipv6.method がすでに **auto** (デフォルト) に設定されている場合は、この手順をスキップしてください。

- 静的 IPv6 アドレス、ネットワークマスク、デフォルトゲートウェイ、DNS サーバー、および検索ドメインを設定するには、次のように入力します。

```
# nmcli connection modify Internal-LAN ipv6.method manual ipv6.addresses
2001:db8:1::fffe/64 ipv6.gateway 2001:db8:1::fffe ipv6.dns 2001:db8:1::ffbb
ipv6.dns-search example.com
```

7. プロファイルの他の設定をカスタマイズするには、次のコマンドを使用します。

```
# nmcli connection modify <connection-name> <setting> <value>
```

値はスペースまたはセミコロンで引用符で囲みます。

8. プロファイルをアクティブ化します。

```
# nmcli connection up Internal-LAN
```

検証

1. NIC の IP 設定を表示します。

```
# ip address show enp1s0
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
group default qlen 1000
    link/ether 52:54:00:17:b8:b6 brd ff:ff:ff:ff:ff:ff
    inet 192.0.2.1/24 brd 192.0.2.255 scope global noprefixroute enp1s0
       valid_lft forever preferred_lft forever
    inet6 2001:db8:1::fffe/64 scope global noprefixroute
       valid_lft forever preferred_lft forever
```

2. IPv4 デフォルトゲートウェイを表示します。

```
# ip route show default
default via 192.0.2.254 dev enp1s0 proto static metric 102
```

3. IPv6 デフォルトゲートウェイを表示します。

```
# ip -6 route show default
default via 2001:db8:1::ffee dev enp1s0 proto static metric 102 pref medium
```

4. DNS 設定を表示します。

```
# cat /etc/resolv.conf
search example.com
nameserver 192.0.2.200
nameserver 2001:db8:1::ffbb
```

複数の接続プロファイルが同時にアクティブな場合、**nameserver** エントリーの順序は、これらのプロファイルの DNS 優先度の値と接続タイプによって異なります。

5. **ping** ユーティリティーを使用して、このホストがパケットを他のホストに送信できることを確認します。

```
# ping <host-name-or-IP-address>
```

トラブルシューティング

- ネットワークケーブルがホストとスイッチに差し込まれていることを確認します。
- リンク障害がこのホストだけに存在するか、同じスイッチに接続された他のホストにも存在するかを確認します。
- ネットワークケーブルとネットワークインターフェイスが予想どおりに機能していることを確認します。ハードウェア診断手順を実施して、不具合ケーブルとネットワークインターフェイスカードを置き換えます。
- ディスクの設定がデバイスの設定と一致しない場合は、NetworkManager を起動するか再起動して、インメモリー接続を作成することで、デバイスの設定を反映します。この問題を回避する方法および詳細は、NetworkManager サービスの再起動後に、NetworkManager が接続を複製する ソリューションを参照してください。

関連情報

- **nm-settings(5)** man ページ

1.3. NMTUIを使用したイーサネット接続の設定

イーサネット経由でホストをネットワークに接続する場合は、**nmtui** アプリケーションを使用して、テキストベースのユーザーインターフェイスで接続の設定を管理できます。**nmtui** では、グラフィカルインターフェイスを使用せずに、新しいプロファイルの作成や、ホスト上の既存のプロファイルの更新を行います。

注記

nmtui で 以下を行います。

- カーソルキーを使用してナビゲートします。
- ボタンを選択して **Enter** を押します。
- **Space** を使用してチェックボックスをオンまたはオフにします。

前提条件

- 物理または仮想イーサネットネットワークインターフェイスコントローラー (NIC) がサーバーに設定されている。

手順

1. 接続に使用するネットワークデバイス名がわからない場合は、使用可能なデバイスを表示します。

```
# nmcli device status
DEVICE     TYPE      STATE                CONNECTION
enp1s0     ethernet  unavailable          --
...
```

2. **nmtui** を開始します。

```
# nmtui
```

3. **Edit a connection** 選択し、**Enter** を押します。

4. 新しい接続プロファイルを追加するか、既存の接続プロファイルを変更するかを選択します。

   - 新しいプロファイルを作成するには、以下を実行します。

     i. **Add** を押します。

     ii. ネットワークタイプのリストから **Ethernet** を選択し、**Enter** を押します。

   - 既存のプロファイルを変更するには、リストからプロファイルを選択し、**Enter** を押します。

5. オプション: 接続プロファイルの名前を更新します。
   ホストに複数のプロファイルがある場合は、わかりやすい名前を付けると、プロファイルの目的を識別しやすくなります。

6. 新しい接続プロファイルを作成する場合は、ネットワークデバイス名を **connection** フィールドに入力します。

7. 環境に応じて、**IPv4 configuration** および **IPv6 configuration** 領域に IP アドレス設定を設定します。これを行うには、これらの領域の横にあるボタンを押して、次を選択します。

   - この接続に IP アドレスが必要ない場合は、**Disabled** にします。

   - DHCP サーバーが IP アドレスをこの NIC に動的に割り当てる場合は、**Automatic** にします。

   - ネットワークで静的 IP アドレス設定が必要な場合は、**Manual** にします。この場合、さらにフィールドに入力する必要があります。

     i. 設定するプロトコルの横にある **Show** を押して、追加のフィールドを表示します。

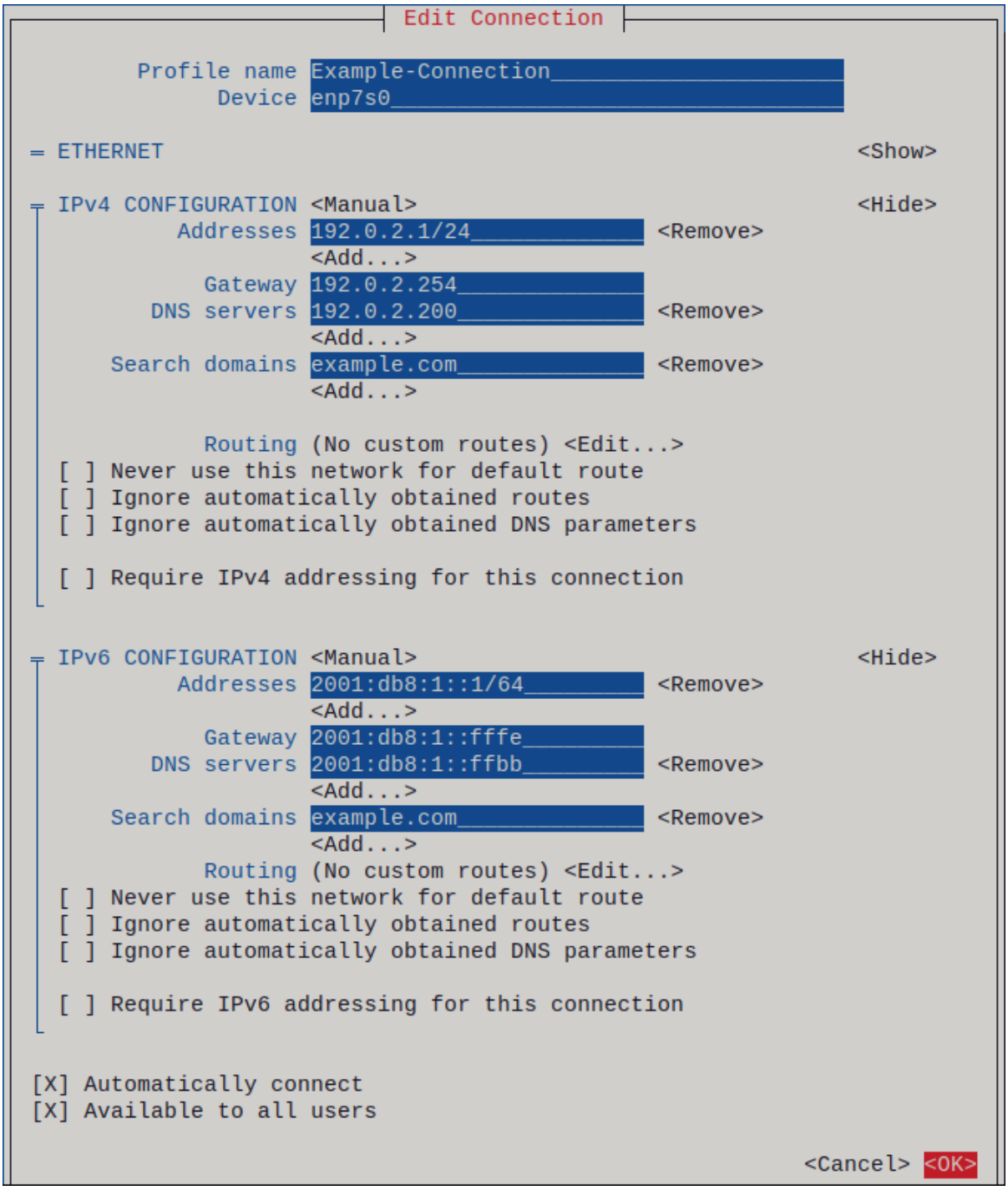
     ii. **Addresses** の横にある **Add** を押して、IP アドレスとサブネットマスクを Classless Inter-Domain Routing (CIDR) 形式で入力します。
     サブネットマスクを指定しない場合、NetworkManager は IPv4 アドレスに **/32** サブネットマスクを設定し、IPv6 アドレスに **/64** サブネットマスクを設定します。

     iii. デフォルトゲートウェイのアドレスを入力します。

     iv. **DNS servers** の横にある **Add** を押して、DNS サーバーのアドレスを入力します。

     v. **Search domains** の横にある **Add** を押して、DNS 検索ドメインを入力します。

図1.1 静的 IP アドレス設定によるイーサネット接続の例



8. **OK** を押すと、新しい接続が作成され、自動的にアクティブ化されます。
9. **Back** を押してメインメニューに戻ります。
10. **Quit** を選択し、**Enter** キーを押して **nmtui** アプリケーションを閉じます。

検証

1. NIC の IP 設定を表示します。

```
# ip address show enp1s0
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
group default qlen 1000
    link/ether 52:54:00:17:b8:b6 brd ff:ff:ff:ff:ff:ff
```

```
    inet 192.0.2.1/24 brd 192.0.2.255 scope global noprefixroute enp1s0
       valid_lft forever preferred_lft forever
    inet6 2001:db8:1::fffe/64 scope global noprefixroute
       valid_lft forever preferred_lft forever
```

2. IPv4 デフォルトゲートウェイを表示します。

```
# ip route show default
default via 192.0.2.254 dev enp1s0 proto static metric 102
```

3. IPv6 デフォルトゲートウェイを表示します。

```
# ip -6 route show default
default via 2001:db8:1::ffee dev enp1s0 proto static metric 102 pref medium
```

4. DNS 設定を表示します。

```
# cat /etc/resolv.conf
search example.com
nameserver 192.0.2.200
nameserver 2001:db8:1::ffbb
```

   複数の接続プロファイルが同時にアクティブな場合、**nameserver** エントリーの順序は、これらのプロファイルの DNS 優先度の値と接続タイプによって異なります。

5. **ping** ユーティリティーを使用して、このホストがパケットを他のホストに送信できることを確認します。

```
# ping <host-name-or-IP-address>
```

トラブルシューティング

- ネットワークケーブルがホストとスイッチに差し込まれていることを確認します。
- リンク障害がこのホストだけに存在するか、同じスイッチに接続された他のホストにも存在するかを確認します。
- ネットワークケーブルとネットワークインターフェイスが予想どおりに機能していることを確認します。ハードウェア診断手順を実施して、不具合ケーブルとネットワークインターフェイスカードを置き換えます。
- ディスクの設定がデバイスの設定と一致しない場合は、NetworkManager を起動するか再起動して、インメモリー接続を作成することで、デバイスの設定を反映します。この問題を回避する方法および詳細は、NetworkManager サービスの再起動後に、NetworkManager が接続を複製する ソリューションを参照してください。

関連情報

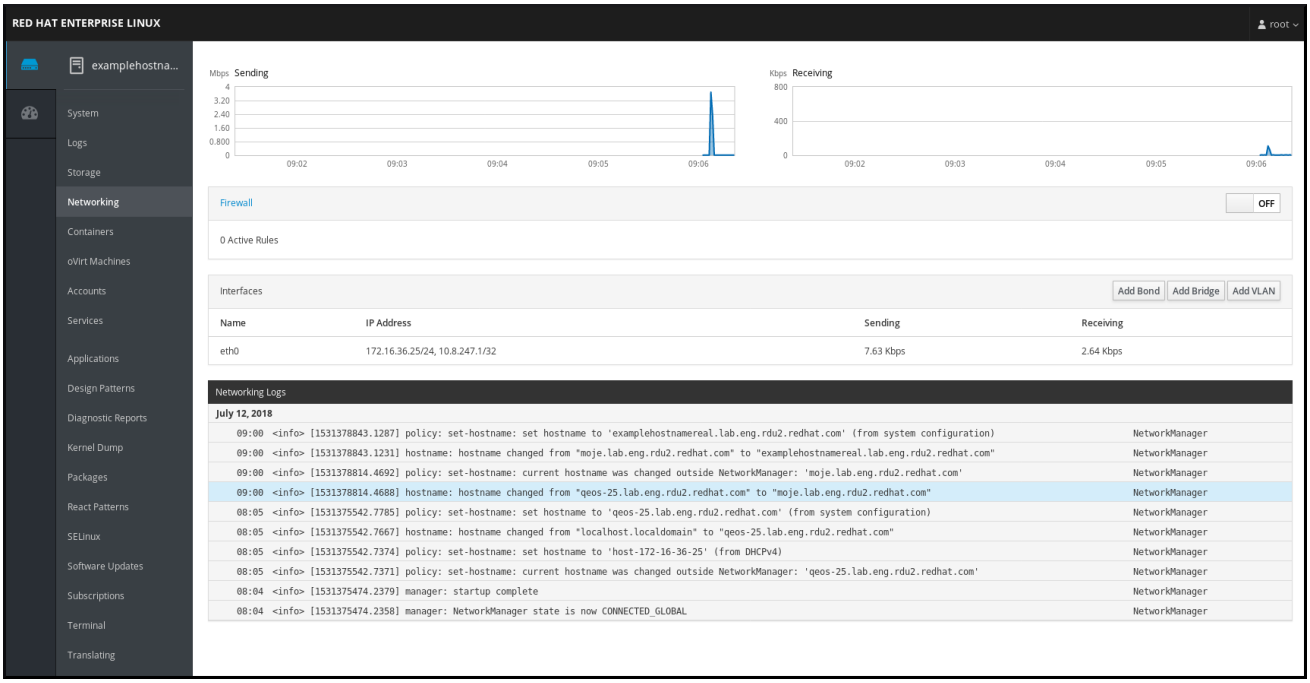
- 特定のプロファイルでのデフォルトゲートウェイの指定を防ぐための NetworkManager の設定
- DNS サーバーの順序の設定

1.4. RHEL WEB コンソールにおけるネットワークの管理

Web コンソールの **Networking** メニューでは、以下が可能です。

- 最近送受信したパケットの表示
- 利用可能なネットワークインターフェイスの最も重要な特徴の表示
- ネットワーキングログのコンテンツの表示
- ネットワークインターフェイスの様々なタイプ (ボンディング、チーム、ブリッジ、VLAN) の追加

図1.2 RHEL Web コンソールにおけるネットワークの管理



1.5. RHEL システムロールを使用したネットワークの管理

network ロールを使用して、複数のターゲットマシンにネットワーク接続を設定できます。

network ロールでは、以下のタイプのインターフェイスを設定できます。

- イーサネット
- ブリッジ
- ボンディング
- VLAN
- MacVLAN
- Infiniband

各ホストに必要なネットワーク接続は、**network_connections** 変数内にリストとして提供されます。

警告

network ロールは、**network_connections** 変数で指定されているとおりに、ターゲットシステムにあるすべての接続プロファイルを更新または作成します。したがって、そのオプションがそのシステムにのみ存在し、**network_connections** 変数にはない場合、**network** ロールは指定されたプロファイルからオプションを削除します。

以下の例は、必要なパラメーターを持つイーサネット接続が確実に設定されるように、**network** ロールを適用する方法を示しています。

必要なパラメーターでイーサネット接続を設定する network ロールを適用する Playbook の例

```
# 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
```

関連情報

- RHEL システムロールを使用するためのコントロールノードと管理対象ノードの準備

1.6. 関連情報

- ネットワークの設定および管理

第2章 システム登録およびサブスクリプション管理

Red Hat Enterprise Linux オペレーティングシステムと、そこにインストールされている製品は、サブスクリプションの対象となります。

Red Hat コンテンツ配信ネットワーク (CDN) サブスクリプションを使用して、以下を追跡します。

- 登録したシステム
- システムにインストールされている製品
- インストール済みの製品に割り当てられているサブスクリプション

2.1. インストール後のシステムの登録

インストールプロセス中にシステムを登録していない場合は、以下の手順に従って登録します。

前提条件

- Red Hat カスタマーポータルに有効なユーザーアカウントがある。
- Red Hat アカウントの作成 ページを参照してください。
- RHEL システムに使用するアクティブなサブスクリプションがある。
- インストールプロセスの詳細は 標準的な RHEL 9 インストールの実行 を参照してください。

手順

1. ワンステップでシステムを登録し、自動的にサブスクライブします。

```
# 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
```

コマンドを実行すると、Red Hat カスタマーポータルのユーザー名とパスワードの入力を求めるプロンプトが表示されます。

登録プロセスに失敗した場合は、システムを特定のプールに登録できます。これを実行する方法については、以下の手順に従います。

a. 必要なサブスクリプションのプール ID を確認します。

```
# subscription-manager list --available
```

このコマンドは、使用している Red Hat アカウントで利用可能なサブスクリプションをすべて表示します。サブスクリプションごとに、プール ID を含むさまざまな情報が表示されます。

b. **pool_id** を、確認したプール ID に置き換えて、適切なサブスクリプションをシステムに割り当てます。

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

注記

Red Hat Insights にシステムを登録するには、**rhc connect** ユーティリティーを使用できます。リモートホスト設定のセットアップ を参照してください。

関連情報

- カスタマーポータル: 自動アタッチシステム
- カスタマーポータル: 登録および手動サブスクリプション

2.2. WEB コンソールで認証情報を使用してサブスクリプションを登録

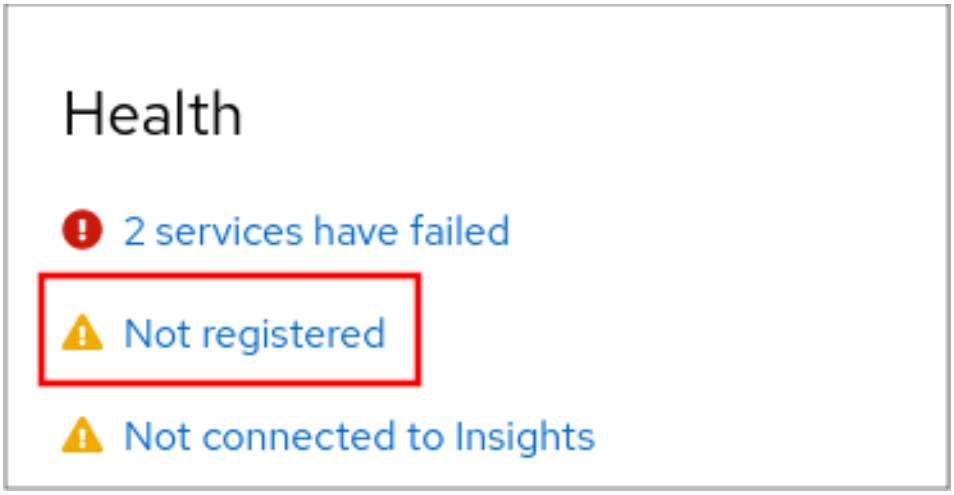
RHEL Web コンソールを使用して、新しくインストールされた Red Hat Enterprise Linux をアカウント認証情報で登録するには、次の手順を使用します。

前提条件

- Red Hat カスタマーポータルに有効なユーザーアカウントがある。
  Red Hat アカウントの作成 ページを参照してください。
- RHEL システムに使用するアクティブなサブスクリプションがある。

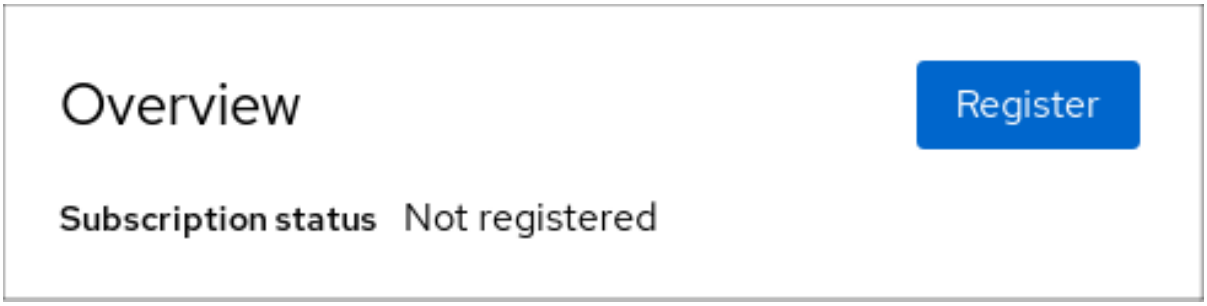
手順

1. RHEL Web コンソールにログインします。詳細は、Web コンソールへのログイン を参照してください。
2. **概要** ページの **ヘルス** ファイル内の **未登録** の警告をクリックするか、メインメニューの **サブスクリプション** をクリックして、サブスクリプション情報のあるページに移動します。

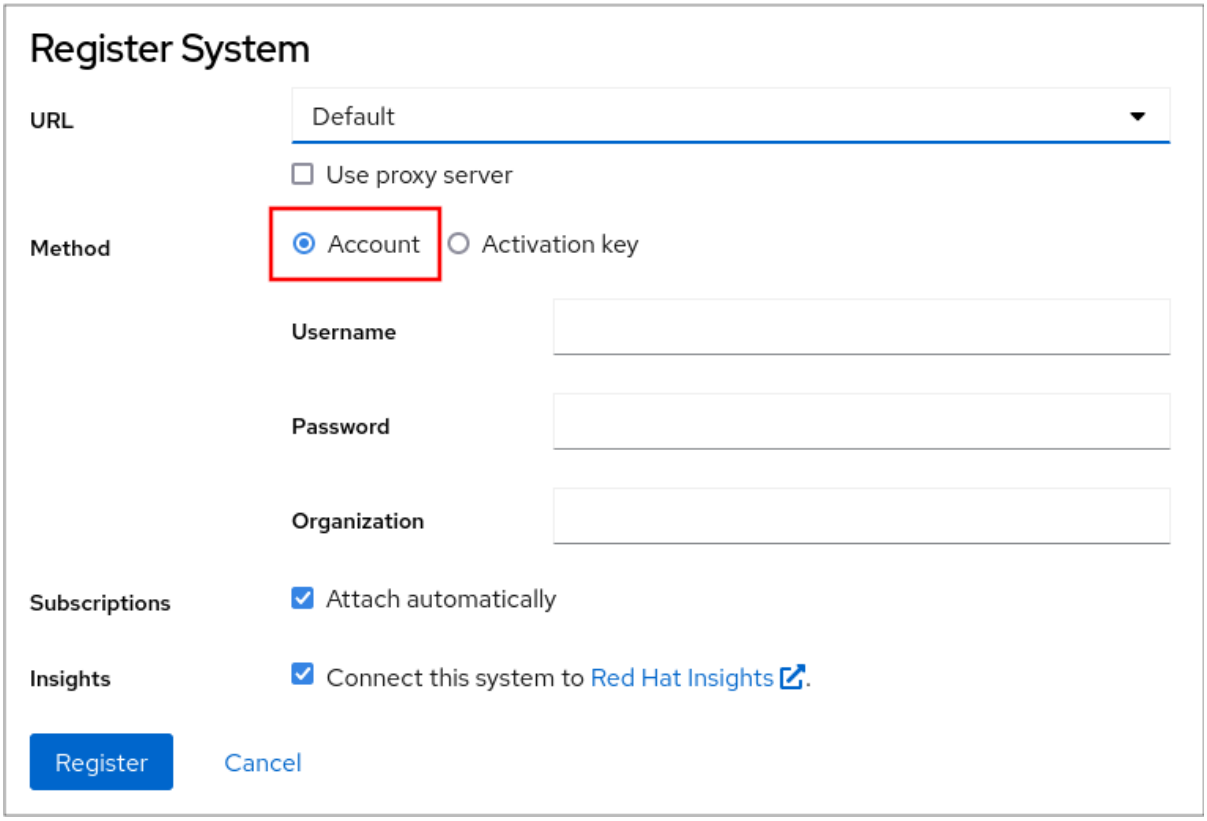


をクリックします。

3. **Overview** フィールドの **Register** をクリックします。



4. **システムの登録** ダイアログボックスで、アカウント情報での登録を選択します。



5. ユーザー名を入力します。
6. パスワードを入力します。
7. オプションで、組織名または ID を入力します。
   アカウントが Red Hat カスタマーポータルで複数の組織に所属している場合には、組織名または組織 ID を追加する必要があります。組織 ID は、Red Hat の連絡先に問い合わせてください。

   - Red Hat Insights にシステムを接続しない場合は、**Insights** チェックボックスのチェックを外してください。

8. **登録** ボタンをクリックします。

この時点で、Red Hat Enterprise Linux システムが正常に登録されました。

2.3. GNOME での RED HAT アカウントを使用したシステム登録

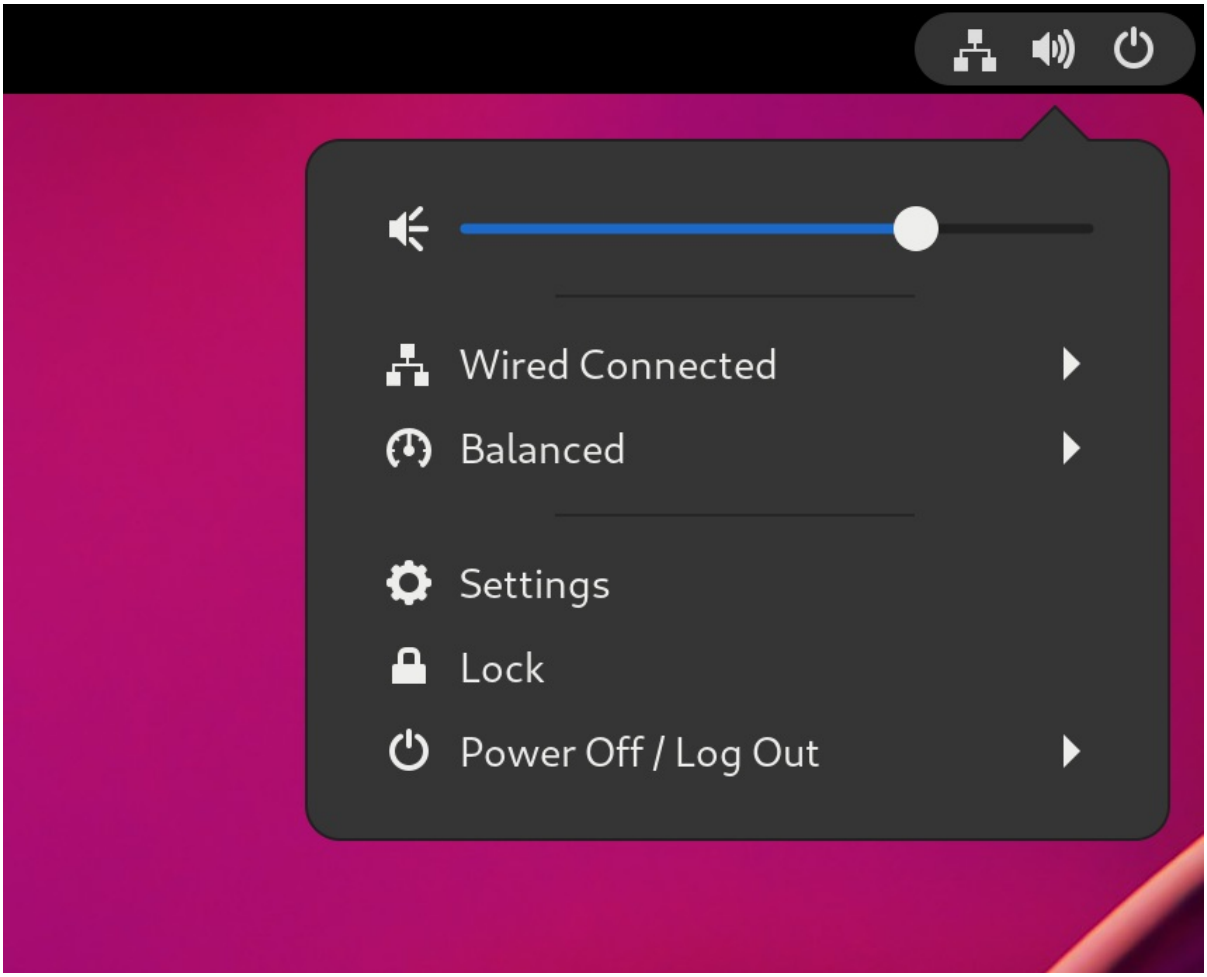
以下の手順に従って、システムを Red Hat アカウントに登録します。

前提条件

- Red Hat カスタマーポータルで有効なアカウント
  新規ユーザー登録は、Red Hat アカウントの作成 ページを参照してください。

手順

1. 画面の右上隅からアクセスできる **システムメニュー** を開き、**設定** をクリックします。



2. **サブスクリプション+について** に移動します。
3. Red Hat サーバーを使用していない場合:
   a. **登録サーバー** セクションで、**カスタムアドレス** を選択します。
   b. **URL** フィールドにサーバーアドレスを入力します。
4. **登録タイプ** セクションで、**Red Hat アカウント** を選択します。
5. **登録の詳細** セクションで:
   - **Login** フィールドに、Red Hat アカウントのユーザー名を入力します。
   - **Password** フィールドに、Red Hat アカウントのパスワードを入力します。
   - **Organizaiton** フィールドに組織の名前を入力します。
6. **Register** をクリックします。

2.4. GNOME でのアクティベーションキーを使用したシステム登録

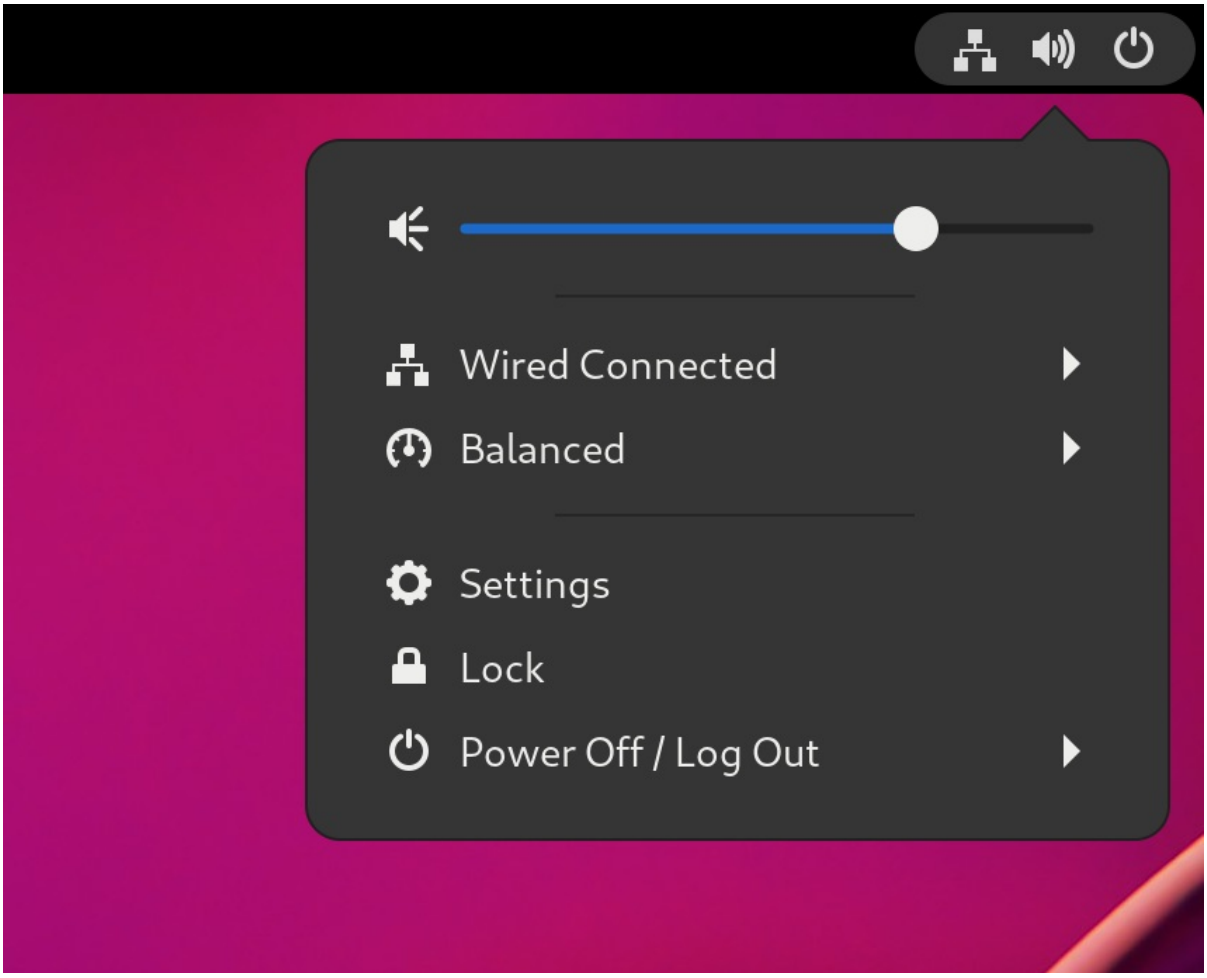
以下の手順に従って、システムをアクティベーションキーに登録します。組織の管理者からアクティベーションキーを取得できます。

前提条件

- アクティベーションキーまたはキー。
  新しい アクティベーションキー を作成するには、アクティベーションキーページを参照してください。

手順

1. 画面の右上隅からアクセスできる **システムメニュー** を開き、**設定** をクリックします。



2. **サブスクリプション+について** に移動します。
3. Red Hat サーバーを使用していない場合:
   a. **登録サーバー** セクションで、**カスタムアドレス** を選択します。
   b. **URL** フィールドにサーバーアドレスを入力します。
4. **登録の種類** セクションで、**アクティベーションキー** を選択します。
5. **Registration Details** で以下を行います。
   - **アクティベーションキーフィールドにアクティベーション** キーを入力します。
     キーをコンマ (,) で区切ります。
   - **組織** フィールドに組織の名前または ID を入力します。

6. **Register** をクリックします。

第3章 RED HAT サポートへのアクセス

本セクションでは、Red Hat サポートおよび **sosreport** を使用して問題を効果的にトラブルシューティングする方法を説明します。

Red Hat サポートを利用する場合は、Red Hat カスタマーポータル にアクセスしてください。カスタマーポータルでは、サブスクリプションで利用可能なものをすべて提供します。

3.1. RED HAT カスタマーポータルで利用できる RED HAT サポート

以下のセクションでは、Red Hat カスタマーポータルを使用してサポートを受ける方法を説明します。

前提条件

- Red Hat カスタマーポータルに有効なユーザーアカウントがある。Red Hat ログインの作成 を参照してください。
- RHEL システムに使用するアクティブなサブスクリプションがある。

手順

1. Red Hat サポート にアクセスします。
   a. サポートケースを作成する
   b. Red Hat 専門スタッフとのライブチャットを開始する
   c. 電話または電子メールで Red Hat 専門スタッフに問い合わせる

3.2. SOSREPORT を使用した問題のトラブルシューティング

sosreport コマンドは設定の詳細、システム情報、および診断情報を Red Hat Enterprise Linux システムから収集します。

次のセクションでは、**sosreport** コマンドを使用して、サポートケースのレポートを作成する方法を説明します。

前提条件

- Red Hat カスタマーポータルに有効なユーザーアカウントがある。Red Hat ログインの作成 を参照してください。
- RHEL システムに使用するアクティブなサブスクリプションがある。
- サポートケース番号。

手順

1. **sos** パッケージをインストールするには、以下のコマンドを実行します。

```
# dnf install sos
```

注記

Red Hat Enterprise Linux のデフォルトの最小インストールには、**sosreport** コマンドを提供する **sos** パッケージは含まれません。

2. レポートを生成します。

```
# sosreport
```

3. サポートケースにレポートを添付します。
   How can I attach a file to a Red Hat support case? を参照してください。詳細は、Red Hat ナレッジベースの記事を参照してください。

   レポートを添付すると、該当のサポートケース番号の入力が求められます。

関連情報

- What is an sosreport and how to create one in Red Hat Enterprise Linux?

第4章 基本的な環境設定の変更

基本的な環境設定は、インストールプロセスの一部です。以下のセクションでは、後での変更する時に説明します。環境の基本設定には、以下が含まれます。

- 日付と時刻
- システムロケール
- キーボードのレイアウト
- 言語

4.1. 日付および時刻の設定

正確な時間管理は、いくつかの理由で重要です。Red Hat Enterprise Linux では、**NTP** プロトコルにより、時刻が管理されます。これは、デーモンにより、ユーザー領域に実装されています。ユーザー領域のデーモンは、カーネルで実行しているシステムクロックを更新します。システムクロックは、さまざまなクロックソースを使用して時間を維持します。

Red Hat Enterprise Linux 9 以降のバージョンでは、**chronyd** デーモンを使用して **NTP** を実装します。**chronyd** は、**chrony** パッケージから入手できます。詳細は、Chrony スイートを使用した NTP の設定 を参照してください。

4.1.1. システムの現在日時の表示

現在の日時を表示するには、以下のいずれかの手順を行います。

手順

1. **date** コマンドを実行します。

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

2. 詳細は、**timedatectl** コマンドを使用して確認します。

```
$ 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
```

関連情報

- Web コンソールでの時間設定の設定
- **man date(1)** および **man timedatectl(1)**

4.2. システムロケールの設定

システム全体にわたるロケール設定は **/etc/locale.conf** ファイルに保存され、システム起動の初期段階で **systemd** デーモンにより読み込まれます。**/etc/locale.conf** に設定したロケール設定は、個別のプログラムやユーザーが上書きしない限り、すべてのサービスやユーザーに継承されます。

手順

- 利用可能なシステムロケール設定をリスト表示するには、次のコマンドを実行します。

```
$ localectl list-locales
C.utf8
aa_DJ
aa_DJ.iso88591
aa_DJ.utf8
...
```

- システムロケール設定の現在のステータスを表示するには、次のコマンドを実行します。

```
$ localectl status
```

- デフォルトのシステムロケールオプションを設定または変更するには、**root** ユーザーで **localectl set-locale** サブコマンドを使用します。以下に例を示します。

```
# localectl set-locale LANG=en_US
```

関連情報

- **man localectl(1)**、**man locale(7)**、および **man locale.conf(5)**

4.3. キーボードレイアウトの設定

キーボードレイアウト設定では、テキストコンソールとグラフィカルユーザーインターフェイスで使用するレイアウトを管理します。

手順

- 利用可能なキーマップをリスト表示するには、以下を実行します。

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

- キーマップ設定の現在のステータスを表示するには、次のコマンドを実行します。

```
$ localectl status
...
VC Keymap: us
...
```

- デフォルトのシステムキーマップを設定または変更します。以下に例を示します。

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

関連情報

- **man localectl(1)**、**man locale(7)**、および **man locale.conf(5)**

4.4. テキストコンソールモードでフォントサイズの変更

setfont コマンドを使用して、仮想コンソールのフォントサイズを変更できます。

- フォント名を指定した **setfont** コマンドを実行します。以下に例を示します。

```
# setfont /usr/lib/kbd/consolefonts/LatArCyrHeb-19.psfu.gz
```

注記

setfont コマンドは、デフォルトで複数のハードコードされたパスを検索します。したがって、**setfont** では、フォントの完全な名前とパスは必要ありません。

- フォントのサイズを水平方向と垂直方向に 2 倍にするには、**-d** パラメーターを指定して **setfont** コマンドを入力します。

```
# setfont -d LatArCyrHeb-16
```

注記

2 倍にできる最大フォントサイズは 16 x 16 ピクセルです。

- システムの再起動中に選択したフォントを保持するには、**/etc/vconsole.conf** ファイルの **FONT** 変数を使用します。次に例を示します。

```
# cat /etc/vconsole.conf
KEYMAP="us"
FONT="eurlatgr"
```

- `kbd` パッケージとともにインストールされる **kbd-misc** パッケージには、さまざまなフォントが含まれています。たとえば、フォント **LatArCyrHeb** には多くのバリアントがあります。

```
# rpm -ql kbd-misc | grep LatAr

/usr/lib/kbd/consolefonts/LatArCyrHeb-08.psfu.gz
/usr/lib/kbd/consolefonts/LatArCyrHeb-14.psfu.gz
/usr/lib/kbd/consolefonts/LatArCyrHeb-16+.psfu.gz
/usr/lib/kbd/consolefonts/LatArCyrHeb-16.psfu.gz
/usr/lib/kbd/consolefonts/LatArCyrHeb-19.psfu.gz
```

注記

仮想コンソールでサポートされる最大フォントサイズは 32 ピクセルです。コンソールの解像度を小さくすることで、フォントの読みやすさの問題を軽減できます。

4.5. 関連情報

- 標準的な RHEL 9 インストールの実行

第5章 2 台のシステム間で OPENSSH を使用した安全な通信の使用

SSH (Secure Shell) は、クライアント/サーバーアーキテクチャーを使用する 2 つのシステム間で安全な通信を提供し、ユーザーがリモートでサーバーホストシステムにログインできるようにするプロトコルです。FTP、Telnet などの他のリモート通信プロトコルとは異なり、SSH はログインセッションを暗号化するため、侵入者が接続して暗号化されていないパスワードを入手するのが困難になります。

Red Hat Enterprise Linux には、基本的な **OpenSSH** パッケージ (一般的な **openssh** パッケージ、**openssh-server** パッケージ、および **openssh-clients** パッケージ) が含まれます。**OpenSSH** パッケージには、**OpenSSL** パッケージ (**openssl-libs**) が必要です。このパッケージは、重要な暗号化ライブラリーをいくつかインストールして、暗号化通信を提供する **OpenSSH** を有効にします。

5.1. SSH と OPENSSH

SSH (Secure Shell) は、リモートマシンにログインしてそのマシンでコマンドを実行するプログラムです。SSH プロトコルは、安全でないネットワーク上で、信頼されていないホスト間で安全な通信を提供します。また、X11 接続と任意の TCP/IP ポートを安全なチャンネルで転送することもできます。

SSH プロトコルは、リモートシェルのログインやファイルコピー用に使用する場合に、システム間の通信の傍受や特定ホストの偽装など、セキュリティーの脅威を軽減します。これは、SSH クライアントとサーバーがデジタル署名を使用してそれぞれの ID を確認するためです。さらに、クライアントシステムとサーバーシステムとの間の通信はすべて暗号化されます。

ホストキーは、SSH プロトコルのホストを認証します。ホスト鍵は、OpenSSH の初回インストール時、またはホストの初回起動時に自動的に生成される暗号鍵です。

OpenSSH は、Linux、UNIX、および同様のオペレーティングシステムでサポートされている SSH プロトコルの実装です。OpenSSH クライアントとサーバー両方に必要なコアファイルが含まれます。OpenSSH スイートは、以下のユーザー空間ツールで構成されます。

- **SSH** は、リモートログインプログラム (SSH クライアント) です。
- **sshd** は、OpenSSH SSH デーモンです。
- **scp** は、安全なリモートファイルコピープログラムです。
- **sftp** は、安全なファイル転送プログラムです。
- **ssh-agent** は、秘密鍵をキャッシュする認証エージェントです。
- **ssh-add** は、秘密鍵の ID を **ssh-agent** に追加します。
- **ssh-keygen** が、**ssh** の認証キーを生成、管理、および変換します。
- **ssh-copy-id** は、ローカルの公開鍵をリモート SSH サーバーの **authorized_keys** ファイルに追加するスクリプトです。
- **ssh-keyscan** - SSH パブリックホストキーを収集します。

注記

RHEL 9 では、Secure copy protocol (SCP) がデフォルトで SSH File Transfer Protocol (SFTP) に置き換えられています。これは、CVE-2020-15778 など、SCP が原因のセキュリティーの問題が発生しているためです。

使用しているシナリオで SFTP が利用できない場合や互換性がない場合は、**-O** オプションを使用して、元の SCP/RCP プロトコルを強制的に使用できます。

追加情報は Red Hat Enterprise Linux 9 の記事の OpenSSH SCP プロトコルが非推奨に を参照してください。

現在、SSH のバージョンには、バージョン 1 と新しいバージョン 2 の 2 つがあります。RHEL の OpenSSH スイートは、SSH バージョン 2 のみをサポートします。このスイートは、バージョン 1 で知られているエクスプロイトに対して脆弱ではない拡張キー交換アルゴリズムを備えています。

RHEL コア暗号化サブシステムの 1 つである OpenSSH は、システム全体の暗号化ポリシーを使用します。これにより、弱い暗号スイートおよび暗号化アルゴリズムがデフォルト設定で無効になります。ポリシーを変更するには、管理者が **update-crypto-policies** コマンドを使用して設定を調節するか、システム全体の暗号化ポリシーを手動でオプトアウトする必要があります。

OpenSSH スイートは、2 セットの設定ファイルを使用します。1 つはクライアントプログラム (つまり、**ssh**、**scp**、および **sftp**) 用で、もう 1 つはサーバー (**sshd** デーモン) 用です。

システム全体の SSH 設定情報が **/etc/ssh/** ディレクトリーに保存されます。ユーザー固有の SSH 設定情報は、ユーザーのホームディレクトリーの **~/.ssh/** に保存されます。OpenSSH 設定ファイルの詳細なリストは、**sshd (8)** の man ページの **FILES** セクションを参照してください。

関連情報

- **man -k ssh** コマンドを使用してリスト表示される man ページ
- システム全体の暗号化ポリシーの使用

5.2. OPENSSH サーバーの設定および起動

お使いの環境と OpenSSH サーバーの起動に必要となる基本設定には、以下の手順を使用します。デフォルトの RHEL インストールを行うと、**sshd** デーモンがすでに起動し、サーバーのホスト鍵が自動的に作成されることに注意してください。

前提条件

- **openssh-server** パッケージがインストールされている。

手順

1. 現行セッションで **sshd** デーモンを開始し、ブート時に自動的に起動するように設定します。

```
# systemctl start sshd
# systemctl enable sshd
```

2. デフォルトの**0.0.0.0** (IPv4) または **::** とは異なるアドレスを指定するには、以下を行います。(IPv6) **/etc/ssh/sshd_config** 設定ファイルの **ListenAddress** ディレクティブ、および低速な動的ネットワーク設定を使用するには、**network-online.target** ターゲットユニットの依存関係

を**sshd.service** ユニットファイルに追加します。これを行うには、以下の内容で **/etc/systemd/system/sshd.service.d/local.conf** ファイルを作成します。

```
[Unit]
Wants=network-online.target
After=network-online.target
```

3. **/etc/ssh/sshd_config** 設定ファイルの OpenSSH サーバーの設定がシナリオの要件を満たしているかどうかを確認します。

4. 必要に応じて、**/etc/issue** ファイルを編集して、クライアント認証を行う前に OpenSSH サーバーに表示される welcome メッセージを変更します。以下に例を示します。

```
Welcome to ssh-server.example.com
Warning: By accessing this server, you agree to the referenced terms and conditions.
```

Banner オプションが **/etc/ssh/sshd_config** でコメントアウトされておらず、その値に **/etc/issue** が含まれていることを確認します。

```
# less /etc/ssh/sshd_config | grep Banner
Banner /etc/issue
```

ログインに成功すると表示されるメッセージを変更するには、サーバーの **/etc/motd** ファイルを編集する必要があります。詳細は、**pam_motd** の man ページを参照してください。

5. **systemd** 設定を再読み込みし、**sshd** を再起動して変更を適用します。

```
# systemctl daemon-reload
# systemctl restart sshd
```

検証

1. **sshd** デーモンが実行していることを確認します。

```
# 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. SSH クライアントを使用して SSH サーバーに接続します。

```
# ssh user@ssh-server-example.com
ECDSA key fingerprint is SHA256:dXbaS0RG/UzlTTku8GtXSz0S1++lPegSy31v3L/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:
```

関連情報

- **sshd(8)** および **sshd_config(5)** の man ページ。

5.3. 鍵ベースの認証用の OPENSSH サーバーの設定

システムのセキュリティーを強化するには、OpenSSH サーバーでパスワード認証を無効にして鍵ベースの認証を有効にします。

前提条件

- **openssh-server** パッケージがインストールされている。
- サーバーで **sshd** デーモンが実行している。

手順

1. テキストエディターで **/etc/ssh/sshd_config** 設定を開きます。以下に例を示します。

```
# vi /etc/ssh/sshd_config
```

2. **PasswordAuthentication** オプションを **no** に変更します。

```
PasswordAuthentication no
```

新しいデフォルトインストール以外のシステムで **PubkeyAuthentication no** が設定されていないことと、**KbdInteractiveAuthentication** ディレクティブが **no** に設定されていることを確認します。リモートで接続している場合は、コンソールもしくは帯域外アクセスを使用せず、パスワード認証を無効にする前に、鍵ベースのログインプロセスをテストします。

3. NFS がマウントされたホームディレクトリーで鍵ベースの認証を使用するには、SELinux ブール値 **use_nfs_home_dirs** を有効にします。

```
# setsebool -P use_nfs_home_dirs 1
```

4. **sshd** デーモンを再読み込みし、変更を適用します。

```
# systemctl reload sshd
```

関連情報

- **sshd(8)**、**sshd_config(5)**、および **setsebool(8)** の man ページ。

5.4. SSH 鍵ペアの生成

以下の手順を使用して、ローカルシステムに SSH 鍵ペアを生成し、生成された公開鍵を OpenSSH サーバーにコピーします。サーバーが正しく設定されている場合は、パスワードなしで OpenSSH サーバーにログインできます。

重要

root で次の手順を完了すると、鍵を使用できるのは **root** だけとなります。

手順

1. SSH プロトコルのバージョン 2 用の ECDSA 鍵ペアを生成するには、次のコマンドを実行します。

```
$ 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+SqwBKRNaU72oZfaCI
joesec@localhost.example.com
The key's randomart image is:
+---[ECDSA 256]---+
|.oo..o=++        |
|.. o .oo .       |
|. .. o. o        |
|....o.+...       |
|o.oo.o +S .      |
|.=.+.   .o       |
|E.*+.  .  . .    |
|.=..+ +..  o     |
|  .  oo*+o.      |
+----[SHA256]-----+
```

ssh-keygen コマンドまたは Ed25519 鍵ペアに **-t rsa** オプションを指定して RSA 鍵ペアを生成するには、**ssh-keygen -t ed25519** コマンドを実行します。

2. 公開鍵をリモートマシンにコピーするには、次のコマンドを実行します。

```
$ 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.
```

セッションで **ssh-agent** プログラムを使用しない場合は、上記のコマンドで、最後に変更した **~/.ssh/id*.pub** 公開鍵をコピーします (インストールされていない場合)。別の公開キーファイルを指定したり、**ssh-agent** により、メモリーにキャッシュされた鍵よりもファイル内の鍵の方が優先順位を高くするには、**-i** オプションを指定して **ssh-copy-id** コマンドを使用します。

注記

システムを再インストールする際に、生成しておいた鍵ペアを引き続き使用する場合は、~/**.ssh**/ ディレクトリーのバックアップを作成します。再インストール後に、このディレクトリーをホームディレクトリーにコピーします。これは、(**root** を含む) システムの全ユーザーで実行できます。

検証

1. パスワードなしで OpenSSH サーバーにログインします。

```
$ ssh joesec@ssh-server-example.com
Welcome message.
...
Last login: Mon Nov 18 18:28:42 2019 from ::1
```

関連情報

- **ssh-keygen (1)** および **ssh-copy-id (1)** の man ページ

5.5. スマートカードに保存された SSH 鍵の使用

Red Hat Enterprise Linux では、OpenSSH クライアントでスマートカードに保存されている RSA 鍵および ECDSA 鍵を使用できるようになりました。この手順に従って、パスワードの代わりにスマートカードを使用した認証を有効にします。

前提条件

- クライアントで、**opensc** パッケージをインストールして、**pcscd** サービスを実行している。

手順

1. PKCS #11 の URI を含む OpenSC PKCS #11 モジュールが提供する鍵のリストを表示し、その出力を **keys.pub** ファイルに保存します。

```
$ ssh-keygen -D pkcs11: > keys.pub
$ ssh-keygen -D pkcs11:
ssh-rsa AAAAB3NzaC1yc2E...KKZMzcQZzx
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. リモートサーバー (**example.com**) でスマートカードを使用した認証を有効にするには、公開鍵をリモートサーバーに転送します。前の手順で作成された **keys.pub** で **ssh-copy-id** コマンドを使用します。

```
$ ssh-copy-id -f -i keys.pub username@example.com
```

3. 手順 1 の **ssh-keygen -D** コマンドの出力にある ECDSA 鍵を使用して **example.com** に接続するには、鍵を一意に参照する URI のサブセットのみを使用できます。以下に例を示します。

```
$ ssh -i "pkcs11:id=%01?module-path=/usr/lib64/pkcs11/opensc-pkcs11.so" example.com
Enter PIN for 'SSH key':
[example.com] $
```

4. ~/**.ssh**/**config** ファイルで同じ URI 文字列を使用して、設定を永続化できます。

```
$ 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] $
```

OpenSSH は **p11-kit-proxy** ラッパーを使用し、OpenSC PKCS #11 モジュールが PKCS#11 キットに登録されているため、以前のコマンドを簡素化できます。

```
$ ssh -i "pkcs11:id=%01" example.com
Enter PIN for 'SSH key':
[example.com] $
```

PKCS #11 の URI の **id=** の部分を飛ばすと、OpenSSH が、プロキシーモジュールで利用可能な鍵をすべて読み込みます。これにより、必要な入力の量を減らすことができます。

```
$ ssh -i pkcs11: example.com
Enter PIN for 'SSH key':
[example.com] $
```

関連情報

- Fedora 28:Better smart card support in OpenSSH
- **p11-kit(8)**、**opensc.conf(5)**、**pcscd(8)**、**ssh(1)**、および **ssh-keygen(1)** の man ページ

5.6. OPENSSH のセキュリティーの強化

以下のヒントは、OpenSSH を使用する際にセキュリティーを高めるのに役に立ちます。OpenSSH 設定ファイル **/etc/ssh/sshd_config** を変更するには、**sshd** デーモンを再読み込みして有効にする必要があることに注意してください。

```
# systemctl reload sshd
```

重要

ほとんどのセキュリティー強化の設定変更により、最新のアルゴリズムまたは暗号スイートに対応していないクライアントとの互換性が低下します。

安全ではない接続プロトコルの無効化

- SSH を本当の意味で有効なものにするため、OpenSSH スイートに置き換えられる安全ではない接続プロトコルを使用しないようにします。このような接続プロトコルを使用すると、ユーザーのパスワード自体は SSH を使用した 1 回のセッションで保護されても、その後に Telnet を使用してログインした時に傍受されてしまうためです。このため、telnet、rsh、rlogin、ftp などの安全ではないプロトコルを無効にすることを検討してください。

鍵ベースの認証の有効化およびパスワードベースの認証の無効化

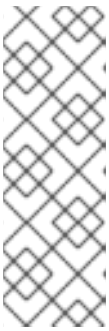
- 認証用パスワードを無効にして鍵のペアのみを許可すると、攻撃対象領域が減ってユーザーの時間を節約できる可能性があります。クライアントにおいて、**ssh-keygen** ツールを使用して鍵のペアを生成し、**ssh-copy-id** ユーティリティーを使用して OpenSSH サーバーのクライアントから公開鍵をコピーします。OpenSSH サーバーでパスワードベースの認証を無効にするには、**/etc/ssh/sshd_config** の **PasswordAuthentication** オプションを **no** に変更します。

```
PasswordAuthentication no
```

鍵のタイプ

- **ssh-keygen** コマンドは、デフォルトで RSA 鍵のペアを生成しますが、**-t** オプションを使用して ECDSA 鍵または Ed25519 鍵を生成するように指定できます。ECDSA (Elliptic Curve Digital Signature Algorithm) は、同等の対称鍵強度で RSA よりも優れたパフォーマンスを提供します。また、短いキーも生成します。Ed25519 公開鍵アルゴリズムは、RSA、DSA、および ECDSA より安全で高速な歪曲エドワーズ曲線の実装です。
サーバーホストの鍵の RSA、ECDSA、および Ed25519 がない場合は、OpenSSH が自動的に作成します。RHEL でホストの鍵の作成を設定するには、インスタンス化したサービス **sshd-keygen@.service** を使用します。たとえば、RSA 鍵タイプの自動作成を無効にするには、次のコマンドを実行します。

```
# systemctl mask sshd-keygen@rsa.service
```



注記

cloud-init が有効になっているイメージでは、**ssh-keygen** ユニットが自動的に無効になります。これは、**ssh-keygen template** サービスが **cloud-init** ツールに干渉し、ホストキーの生成で問題が発生する可能性があるためです。これらの問題を回避するには、**cloud-init** が実行している場合に、**etc/systemd/system/sshd-keygen@.service.d/disable-sshd-keygen-if-cloud-init-active.conf** ドロップイン設定ファイルにより **ssh-keygen** ユニットが無効になります。

- SSH 接続の特定の鍵タイプを除外するには、**/etc/ssh/sshd_config** で該当行をコメントアウトして **sshd** サービスを再読み込みします。たとえば、Ed25519 ホストキーだけを許可するには、次のコマンドを実行します。

```
# HostKey /etc/ssh/ssh_host_rsa_key
# HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key
```

重要

Ed25519 アルゴリズムは FIPS-140 に準拠していないため、OpenSSH は FIPS モードの Ed25519 キーでは機能しません。

デフォルト以外のポート

- デフォルトでは、**sshd** デーモンは TCP ポート 22 をリッスンします。ポートを変更すると、自動ネットワークスキャンに基づく攻撃にシステムがさらされる可能性が減るため、あいまいさによりセキュリティーが向上します。ポートは、**/etc/ssh/sshd_config** 設定ファイルの **Port** ディレクティブを使用して指定できます。

また、デフォルト以外のポートを使用できるように、デフォルトの SELinux ポリシーも更新する必要があります。そのためには、**policycoreutils-python-utils** パッケージの **semanage** ツールを使用します。

```
# semanage port -a -t ssh_port_t -p tcp <port_number>
```

さらに、**firewalld** 設定を更新します。

```
# firewall-cmd --add-port <port_number>/tcp
# firewall-cmd --remove-port=22/tcp
# firewall-cmd --runtime-to-permanent
```

前のコマンドの <port_number> は、**Port** ディレクティブを使用して指定した新しいポート番号に置き換えます。

Root ログイン

- **PermitRootLogin** はデフォルトで **prohibit-password** に設定されています。これにより、root としてログインしてパスワードを使用する代わりに鍵ベースの認証が使用され、ブルートフォース攻撃を防ぐことでリスクが軽減します。

警告

root ユーザーとしてログインを有効にすることは、どのユーザーがどの特権コマンドを実行するかを監査できないため、安全ではありません。管理コマンドを使用するには、ログインして、代わりに **sudo** を使用します。

X セキュリティー拡張機能の使用

- Red Hat Enterprise Linux クライアントの X サーバーは、X セキュリティー拡張を提供しません。そのため、クライアントは X11 転送を使用して信頼できない SSH サーバーに接続するときに別のセキュリティー層を要求できません。ほとんどのアプリケーションは、この拡張機能を有効にしても実行できません。
  デフォルトでは、**/etc/ssh/ssh_config.d/50-redhat.conf** ファイルの **ForwardX11Trusted** オプションが **yes** に設定され、**ssh -X remote_machine** コマンド (信頼できないホスト) と **ssh -Y remote_machine** コマンド (信頼できるホスト) には違いがありません。

  シナリオで X11 転送機能を必要としない場合は、**/etc/ssh/sshd_config** 設定ファイルの **X11Forwarding** ディレクティブを **no** に設定します。

特定のユーザー、グループ、またはドメインへのアクセス制限

- **/etc/ssh/sshd_config** 設定ファイルの **AllowUsers** ディレクティブおよび **AllowGroups** ディレクティブを使用すると、特定のユーザー、ドメイン、またはグループのみが OpenSSH サーバーに接続することを許可できます。**AllowUsers** および **AllowGroups** を組み合わせて、アクセスをより正確に制限できます。以下に例を示します。

```
AllowUsers *@192.168.1.* *@10.0.0.* !*@192.168.1.2
AllowGroups example-group
```

この設定行は、192.168.1.* サブネットおよび 10.0.0.* のサブネットのシステムの全ユーザーからの接続を許可します (192.168.1.2 アドレスのシステムを除く)。すべてのユーザーは、**example-group** グループに属している必要があります。OpenSSH サーバーは、その他のすべての接続を拒否します。

OpenSSH サーバーは、**/etc/ssh/sshd_config** 内のすべての Allow および Deny ディレクティブを渡す接続のみを許可します。たとえば、**AllowUsers** ディレクティブに、**AllowGroups** ディレクティブにリストされているグループの一部ではないユーザーがリストされている場合、そのユーザーはログインできません。

許可リストは、許可されていない新しいユーザーまたはグループもブロックするため、許可リスト (Allow で始まるディレクティブ) の使用は、拒否リスト (Deny で始まるオプション) を使用するよりも安全です。

システム全体の暗号化ポリシーの変更

- OpenSSH は、RHEL のシステム全体の暗号化ポリシーを使用し、デフォルトのシステム全体の暗号化ポリシーレベルは、現在の脅威モデルに安全な設定を提供します。暗号化の設定をより厳格にするには、現在のポリシーレベルを変更します。

```
# update-crypto-policies --set FUTURE
Setting system policy to FUTURE
```

警告

システムがインターネット上で通信する場合、**FUTURE** ポリシーの厳密な設定により、相互運用性の問題が発生する可能性があります。

システム全体の暗号化ポリシーにより、SSH プロトコルの特定の暗号のみを無効にすることもできます。詳細は、セキュリティーの強化 ドキュメントの サブポリシーを使用したシステム全体の暗号化ポリシーのカスタマイズ セクションを参照してください。

OpenSSH サーバーのシステム全体の暗号化ポリシーをオプトアウトするには、**/etc/ssh/sshd_config.d/** ディレクトリーにあるドロップイン設定ファイルに暗号化ポリシーを指定します。このとき、辞書式順序で **50-redhat.conf** ファイルよりも前に来るように、50 未満の 2 桁の数字接頭辞と、**.conf** という接尾辞を付けます (例: **49-crypto-policy-override.conf**)。

詳細は、**sshd_config(5)** の man ページを参照してください。

OpenSSH クライアントのシステム全体の暗号化ポリシーをオプトアウトするには、次のいずれかのタスクを実行します。

- 指定のユーザーの場合は、**~/.ssh/config** ファイルのユーザー固有の設定でグローバルの **ssh_config** を上書きします。
- システム全体の場合は、**/etc/ssh/ssh_config.d/** ディレクトリーにあるドロップイン設定ファイルに暗号化ポリシーを指定します。このとき、辞書式順序で **50-redhat.conf** ファイルよりも前に来るように、50 未満の 2 桁の接頭辞と、**.conf** という接尾辞を付けます (例: **49-crypto-policy-override.conf**)。

関連情報

- **sshd_config(5)**、**ssh-keygen(1)**、**crypto-policies(7)**、および **update-crypto-policies(8)** の man ページ
- セキュリティー強化 ドキュメントの システム全体の暗号化ポリシーの使用
- ssh サービスのみに対して特定のアルゴリズムと暗号を無効化する方法 の記事

5.7. SSH ジャンプホストを使用してリモートサーバーに接続

この手順に従って、ジャンプホストとも呼ばれる中間サーバーを介してローカルシステムをリモートサーバーに接続します。

前提条件

- ジャンプホストでローカルシステムからの SSH 接続に対応している。
- リモートサーバーが、ジャンプホストからのみ SSH 接続を受け入れる。

手順

1. ローカルシステムの ~/**.ssh/config** ファイルを編集してジャンプホストを定義します。以下に例を示します。

```
Host jump-server1
  HostName jump1.example.com
```

   - **Host** パラメーターは、**ssh** コマンドで使用できるホストの名前またはエイリアスを定義します。値は実際のホスト名と一致可能ですが、任意の文字列にすることもできます。
   - **HostName** パラメーターは、ジャンプホストの実際のホスト名または IP アドレスを設定します。

2. **ProxyJump** ディレクティブを使用してリモートサーバーのジャンプ設定を、ローカルシステムの ~/**.ssh/config** ファイルに追加します。以下に例を示します。

```
Host remote-server
  HostName remote1.example.com
  ProxyJump jump-server1
```

3. ローカルシステムを使用して、ジャンプサーバー経由でリモートサーバーに接続します。

```
$ ssh remote-server
```

   このコマンドは、設定手順 1 および 2 を省略したときの **ssh -J jump-server1 remote-server** コマンドと同じです。

注記

ジャンプサーバーをさらに指定することもできます。また、完全なホスト名を指定する場合は、設定ファイルへのホスト定義の追加を飛ばすこともできます。以下に例を示します。

```
$ ssh -J jump1.example.com,jump2.example.com,jump3.example.com
remote1.example.com
```

ジャンプサーバーのユーザー名または SSH ポートが、リモートサーバーの名前およびポートと異なる場合は、上記のコマンドのホスト名のみの表記を変更します。以下に例を示します。

```
$ ssh -J
johndoe@jump1.example.com:75,johndoe@jump2.example.com:75,johndoe@ju
mp3.example.com:75 joesec@remote1.example.com:220
```

関連情報

- **ssh_config(5)** および **ssh(1)** の man ページ

5.8. SSH-AGENT を使用して SSH キーでリモートマシンに接続する手順

パスフレーズを SSH 接続を開始するたびに入力しなくて済むようにするには、**ssh-agent** ユーティリティーを使用して SSH 秘密鍵をキャッシュします。秘密鍵とパスフレーズのセキュリティーが確保されます。

前提条件

- SSH デーモンが実行中で、ネットワーク経由で到達可能なリモートホストがある。
- リモートホストにログインするための IP アドレスまたはホスト名および認証情報を把握している。
- パスフレーズで SSH キーペアを生成し、公開鍵をリモートマシンに転送している。

詳細は、SSH 鍵ペアの生成 を参照してください。

手順

1. オプション: キーを使用してリモートホストに対して認証できることを確認します。
   a. SSH を使用してリモートホストに接続します。

   ```
   $ ssh example.user1@198.51.100.1 hostname
   ```

   b. 秘密鍵へのアクセス権を付与する鍵の作成時に指定したパスフレーズを入力します。

   ```
   $ ssh example.user1@198.51.100.1 hostname
    host.example.com
   ```

2. **ssh-agent** を起動します。

```
$ eval $(ssh-agent)
Agent pid 20062
```

3. **ssh-agent** にキーを追加します。

```
$ ssh-add ~/.ssh/id_rsa
Enter passphrase for ~/.ssh/id_rsa:
Identity added: ~/.ssh/id_rsa (example.user0@198.51.100.12)
```

検証

- オプション: SSH を使用してホストマシンにログインします。

```
$ ssh example.user1@198.51.100.1

Last login: Mon Sep 14 12:56:37 2020
```

パスフレーズを入力する必要がないことに注意してください。

5.9. 関連情報

- **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)**、および **crypto-policies(7)** の man ページ
- OpenSSH のホームページ
- 非標準設定でのアプリケーションとサービスの SELinux 設定
- Controlling network traffic using firewalld

第6章 基本的なシステムセキュリティーの設定

コンピューターセキュリティーとは、盗難、損傷、破壊、および誤りからコンピューターシステムやハードウェア、ソフトウェア、情報、およびサービスを保護することです。機密データを処理してビジネス取引を扱う企業では特に、コンピューターセキュリティーの確保は必須タスクです。

本セクションでは、オペレーティングシステムのインストール後に設定できる基本的なセキュリティー機能のみを説明します。

6.1. ファイアウォールサービスの有効化

ファイアウォールは、デフォルトのセキュリティールールに基づいてネットワークトラフィックの送受信の監視および制御を行うネットワークセキュリティーシステムです。ファイアウォールは、通常、信頼できる安全な内部ネットワークと、その他の外部ネットワークとの間に壁を作ります。

Red Hat Enterprise Linux でファイアウォールを提供する **firewalld** サービスは、インストール時に自動的に有効になります。

firewalld サービスを有効にするには、以下の手順に従います。

手順

- **firewalld** の現在の状況の表示

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

- **firewalld** が有効になっていない場合は、**root** ユーザーに切り替えて、**firewalld** サービスを起動し、システムの再起動後に自動的に起動できるようにします。

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

検証手順

- **firewalld** が実行中で、有効になっていることを確認します。

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

関連情報

- firewalld の使用および設定
- **man firewalld(1)**

6.2. 基本的な SELINUX 設定の管理

Security Enhanced Linux (SELinux) は、どのプロセスがどのファイル、ディレクトリー、およびポートにアクセスできるのかを指定するシステムセキュリティーの追加レイヤーです。これらのパーミッションは、SELinux ポリシーで定義します。ポリシーは、SELinux セキュリティーエンジンをガイドする一連のルールです。

SELinux のステータスには、以下の 2 つがあります。

- 無効
- 有効

SELinux が有効な場合は、以下のいずれのモードで実行できます。

- 有効
  - Enforcing
  - Permissive

Enforcing モード では、SELinux は読み込まれたポリシーを強制的に実行します。SELinux は、SELinux ポリシールールに基づいてアクセスを拒否し、明示的に許可された対話だけを有効にします。Enforcing モードは最も安全な SELinux モードであり、インストール後のデフォルトモードです。

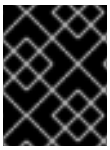
Permissive モード では、SELinux は読み込まれたポリシーを強制に実行しません。SELinux はアクセスを拒否しませんが、ルールに違反するアクションを **/var/log/audit/audit.log** ログで報告します。Permissive モードは、インストール時のデフォルトのモードです。Permissive モードは、問題のトラブルシューティングなど、特定のケースで役に立ちます。

関連情報

- SELinux の使用

6.3. SELINUX で必要なステータスの確認

デフォルトでは、SELinux は Enforcing モードで動作します。ただし、特定のシナリオでは、SELinux を Permissive モードに設定したり、無効にしたりすることもできます。



重要

Red Hat は、Enforcing モードでシステムを使用することを推奨します。デバッグの目的で、SELinux を Permissive モードに設定します。

以下の手順に従って、システムの SELinux の状態とモードを変更します。

手順

1. 現在有効な SELinux モードを表示します。

```
$ getenforce
```

2. SELinux を一時的に設定します。
   a. Enforcing モードに設定する場合:

```
# setenforce Enforcing
```

b. Permissive モードに設定する場合:

```
# setenforce Permissive
```

注記

再起動後に、SELinux モードは **/etc/selinux/config** 設定ファイルで指定された値に設定されます。

3. 再起動後も SELinux モードを保持するように設定するには、**/etc/selinux/config** 設定ファイルの **SELINUX** 変数を変更します。
たとえば、SELinux を Enforcing モードに切り替えるには、以下のように設定します。

```
# 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
...
```

警告

SELinux を無効にすると、システムセキュリティーが低下します。無効にすると、メモリーリークや競合状態によりカーネルパニックが発生する可能性があるため、**/etc/selinux/config** ファイルの **SELINUX=disabled** オプションを使用して SELinux を無効にしないでください。代わりに、**selinux=0** パラメーターをカーネルコマンドラインに追加して、SELinux を無効にします。詳細は、Changing SELinux modes at boot time を参照してください。

関連情報

- SELinux のステータスおよびモードの変更

6.4. 関連情報

- SSH 鍵ペアの生成
- 鍵ベースの認証用の OpenSSH サーバーの設定
- セキュリティーの強化
- SELinux の使用
- ネットワークのセキュリティー保護

第7章 RHEL システムロールの概要

RHEL システムロールを使用すると、複数の RHEL メジャーバージョンにわたる複数の RHEL システムのシステム設定をリモートで管理できます。

重要な用語と概念

以下では、Ansible 環境における重要な用語と概念を説明します。

コントロールノード

コントロールノードは、Ansible コマンドと Playbook を実行するシステムです。コントロールノードには、Ansible Automation Platform、Red Hat Satellite、または RHEL 9、8、または 7 ホストを使用できます。詳細は、Preparing a control node on RHEL 9 を参照してください。

管理対象ノード

管理対象ノードは、Ansible で管理するサーバーとネットワークデバイスです。管理対象ノードは、ホストと呼ばれることもあります。管理対象ノードに Ansible をインストールする必要はありません。詳細は、管理対象ノードの準備 を参照してください。

Ansible Playbook

Playbook では、管理対象ノード上で実現したい設定、または管理対象ノード上のシステムが実行する一連の手順を定義します。Playbook は、Ansible の設定、デプロイメント、およびオーケストレーションの言語です。

Inventory

インベントリーファイルでは、管理対象ノードをリストし、各管理対象ノードの IP アドレスなどの情報を指定します。インベントリーでは、管理対象ノードを整理し、グループを作成およびネストして、スケーリングを容易にすることもできます。インベントリーファイルは、ホストファイルと呼ばれることもあります。

Red Hat Enterprise Linux 9 コントロールノードで利用可能なロール

Red Hat Enterprise Linux 9 コントロールノードでは、**rhel-system-roles** パッケージが次のロールを提供します。

ロール名	ロールの説明	章のタイトル
certificate	証明書の発行および更新	RHEL システムロールを使用した証明書の要求
cockpit	Web コンソール	Cockpit RHEL システムロールを使用した Web コンソールのインストールと設定
crypto_policies	システム全体の暗号化ポリシー	システム間でのカスタム暗号化ポリシーの設定
ファイアウォール (firewall)	Firewalld	システムロールを使用した firewalld の設定
ha_cluster	HA クラスター	システムロールを使用した高可用性クラスターの設定
kdump	カーネルダンプ	RHEL システムロールを使用した kdump の設定

ロール名	ロールの説明	章のタイトル
kernel_settings	カーネル設定	Ansible ロールを使用したカーネルパラメーターの永続的な設定
logging	Logging	ロギングシステムロールの使用
metrics	メトリック (PCP)	RHEL システムロールを使用したパフォーマンスの監視
network	ネットワーク	network RHEL システムロールを使用した InfiniBand 接続の管理
nbde_client	ネットワークバインドディスク暗号化クライアント	nbde_client および nbde_server システムロールの使用
nbde_server	ネットワークバインドディスク暗号化サーバー	nbde_client および nbde_server システムロールの使用
postfix	postfix	システムロールの postfix ロールの変数
postgresql	PostgreSQL	postgresql RHEL システムロールを使用した PostgreSQL のインストールと設定
selinux	SELinux	システムロールを使用した SELinux の設定
ssh	SSH クライアント	ssh システムロールを使用したセキュアな通信の設定
sshd	SSH サーバー	ssh システムロールを使用したセキュアな通信の設定
ストレージ	Storage	RHEL システムロールを使用したローカルストレージの管理
tlog	ターミナルセッションの記録	tlog RHEL システムロールを使用したセッション記録用システムの設定
timesync	時刻同期	RHEL システムロールを使用した時刻同期の設定
vpn	VPN	vpn RHEL システムロールを使用した IPsec による VPN 接続の設定

関連情報

- RHEL システムロールを使用したシステム管理の自動化
- Red Hat Enterprise Linux (RHEL) system roles

- **/usr/share/ansible/roles/rhel-system-roles.<role_name>/README.md** ファイル
- **/usr/share/doc/rhel-system-roles/<role_name>/** ディレクトリー

第8章 ログファイルを使用した問題のトラブルシューティング

ログファイルは、システム (カーネル、サービス、および実行中のアプリケーションなど) に関するメッセージが含まれるファイルです。ログファイルには、問題のトラブルシューティングやシステム機能の監視に役立つ情報が含まれています。Red Hat Enterprise Linux におけるロギングシステムは、組み込みの **syslog** プロトコルに基づいています。特定のプログラムがこのシステムを使用してイベントを記録し、ログファイルに分類します。 これは、オペレーティングシステムの監査およびさまざまな問題のトラブルシューティングに役立ちます。

8.1. SYSLOG メッセージを処理するサービス

以下の 2 つのサービスは、**syslog** メッセージを処理します。

- **systemd-journald** デーモン
- **Rsyslog** サービス

systemd-journald デーモンは、さまざまなソースからメッセージを収集し、収集したメッセージを処理するために **Rsyslog** に転送します。**systemd-journald** デーモンは、以下のソースからメッセージを収集します。

- カーネル
- ブートプロセスの初期段階
- 起動時および実行時のデーモンの標準出力とエラー
- **Syslog**

Rsyslog サービスは、タイプおよび優先順で **syslog** のメッセージを分類し、**/var/log** ディレクトリー内のファイルに書き込みます。**/var/log** ディレクトリーは、ログメッセージを永続的に保存します。

8.2. SYSLOG メッセージを保存するサブディレクトリー

/var/log ディレクトリー内の以下のサブディレクトリーでは、**syslog** メッセージを保存します。

- **/var/log/messages** - 以下を除くすべての **syslog** メッセージ
- **/var/log/secure** - セキュリティーおよび認証に関連するメッセージおよびエラー
- **/var/log/maillog** - メールサーバーに関連するメッセージおよびエラー
- **/var/log/cron** - 定期的に実行されるタスクに関連するログファイル
- **/var/log/boot.log** - システムの起動に関連するログファイル

8.3. WEB コンソールでログファイルの検査

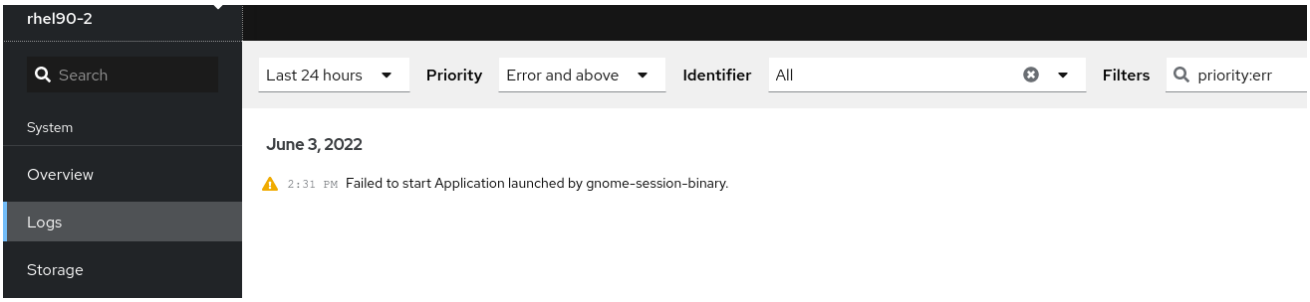
以下の手順に従って、RHEL Web コンソールを使用してログファイルを検証します。

手順

1. RHEL Web コンソールにログインします。詳細はWeb コンソールへのログイン を参照してください。

2. Logs をクリックします。

図8.1 RHEL 9 Web コンソールでログファイルの検査



8.4. コマンドラインでのログの表示

Journal は、ログファイルの表示および管理に役立つ systemd のコンポーネントです。従来のロギングに関連する問題に対応し、残りのシステムと密接に統合され、ログファイルのさまざまなロギングテクノロジーおよびアクセス管理をサポートします。

journalctl コマンドを使用すると、以下のようにコマンドラインを使用してシステムジャーナルのメッセージを表示できます。

```
$ 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
...
```

表8.1 システム情報の表示

コマンド	説明
journalctl	収集されたジャーナルエントリーをすべて表示します。
journalctl FILEPATH	特定のファイルに関連するログを表示します。たとえば、**journalctl /dev/sda** コマンドは、**/dev/sda** ファイルシステムに関連するログを表示します。
journalctl -b	現在のブートのログを表示します。
journalctl -k -b -1	現在のブートのカーネルログを表示します。

表8.2 特定のサービスに関する情報の表示

コマンド	説明
journalctl -b _SYSTEMD_UNIT=<name.service>	ログをフィルターして、**systemd** サービスに一致するエントリーを表示します。

コマンド	説明
journalctl -b _SYSTEMD_UNIT=<name.service> _PID=<number>	一致を組み合わせます。たとえば、このコマンドは、**<name.service>** と PID**<number>** に一致する **systemd-units** のログを表示します。
journalctl -b _SYSTEMD_UNIT=<name.service> _PID=<number> + _SYSTEMD_UNIT=<name2.service>	プラス記号 (+) セパレーターは、論理 OR で 2 つの式を組み合わせます。たとえば、このコマンドは **<name.service>** サービスプロセスからのすべてのメッセージを **PID** で表示し、(プロセスのいずれかの) **<name2.service>** サービスからのすべてのメッセージも表示します。
journalctl -b _SYSTEMD_UNIT=<name.service> _SYSTEMD_UNIT=<name2.service>	このコマンドは、同じフィールドを参照し、いずれかの式に一致するエントリーをすべて表示します。このコマンドは、systemd-unit **<name.service>** または systemd-unit **<name2.service>** に一致するログを表示します。

表8.3 特定のブートに関連するログの表示

コマンド	説明
journalctl --list-boots	ブート番号、その ID、およびブートに関する最初のメッセージと最後のメッセージのタイムスタンプの表形式リストが表示されます。以下のコマンドの ID を使用して詳細情報を表示できます。
journalctl --boot=ID _SYSTEMD_UNIT=<name.service>	指定したブート ID に関する情報を表示します。

8.5. 関連情報

- **man journalctl(1)**
- リモートロギングソリューションの設定

第9章 ユーザーおよびグループの管理

ファイルやプロセスへの不正アクセスを防止するには、ユーザーとグループを正確に管理する必要があります。アカウントを一元管理していない場合、または特定のシステム上でのみユーザーアカウントやグループが必要な場合は、そのホスト上でローカルにユーザーアカウントやグループを作成できます。

9.1. ユーザーアカウントおよびグループアカウントの管理の概要

ユーザーとグループの制御は、Red Hat Enterprise Linux (RHEL) システム管理の中核となる要素です。各 RHEL ユーザーには各種ログイン認証情報があり、さまざまなグループに割り当ててシステム権限をカスタマイズすることができます。

9.1.1. ユーザーとグループの概要

ファイルを作成するユーザーは、そのファイルの所有者 **および** グループ所有者です。ファイルには、所有者、グループ、およびそのグループ外のユーザーに対して読み取り、書き込み、実行のパーミッションが別々に割り当てられます。ファイルの所有者は、**root** ユーザーのみが変更できます。ファイルへのアクセス権限を変更できるのは、**root** ユーザー、ファイル所有者の両方です。通常ユーザーは、所有するファイルのグループ所有権を、所属するグループに変更できます。

各ユーザーは、**ユーザー ID** (**UID**) と呼ばれる一意の数値 ID に関連付けられています。各グループは **グループ ID** (**GID**) に関連付けられています。グループ内のユーザーは、そのグループが所有するファイルの読み取り、書き込み、実行を行う権限を共有します。

9.1.2. 予約ユーザーおよびグループ ID の設定

RHEL は、999 以下のユーザー ID とグループ ID をシステムユーザーとグループ用に予約しています。予約ユーザー ID とグループ ID は、**setup** パッケージで確認できます。予約ユーザー ID とグループ ID を表示するには、以下を使用します。

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

予約範囲は今後増える可能性があるため、新規ユーザーおよびグループには、5000 以降の ID を割り当てることを推奨します。

デフォルトで新規ユーザーに割り当てる ID を 5000 以降に指定するには、**/etc/login.defs** ファイルの **UID_MIN** と **GID_MIN** パラメーターを変更します。

手順

デフォルトで新規ユーザーに割り当てる ID を 5000 以降にするには、以下のコマンドを実行します。

1. 任意のエディターで **/etc/login.defs** ファイルを開きます。
2. UID の自動選択の最小値を定義する行を見つけます。

```
# Min/max values for automatic uid selection in useradd
#
UID_MIN                  1000
```

3. **UID_MIN** の値を 5000 から開始するように変更します。

```
# Min/max values for automatic uid selection in useradd
#
UID_MIN                  5000
```

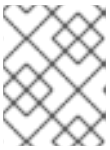
4. GID の自動選択の最小値を定義する行を見つけます。

```
# Min/max values for automatic gid selection in groupadd
#
GID_MIN                  1000
```

5. **GID_MIN** の値を 5000 から開始するように変更します。

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

通常のユーザーに動的に割り当てられる UID と GID は、5000 から始まります。



注記

UID_MIN および GID_MIN の値を変更する前に作成された UID および GID のユーザーおよびグループは変更されません。

これにより、新規ユーザーのグループに UID および GID と同じ 5000+ ID を持たせることができます。

警告

上限が 1000 のシステムとの競合を回避するため、**SYS_UID_MAX** を変更して、システムが予約している ID を 1000 以上にしないようにしてください。

9.1.3. ユーザープライベートグループ

RHEL は、**ユーザープライベートグループ** (**UPG**) システム設定を使用するため、UNIX グループの管理が容易になります。ユーザープライベートグループは、新規ユーザーがシステムに追加されるたびに作成されます。ユーザープライベートグループは作成したユーザーと同じ名前となり、そのユーザーがそのユーザープライベートグループの唯一のメンバーになります。

UPG は、複数ユーザー間のプロジェクトの連携を簡素化します。さらに、UPG のシステム設定では、ユーザーおよびこのユーザーが所属するグループ両方がファイルまたはディレクトリーを変更できるので、新規作成されたファイルまたはディレクトリーのデフォルトの権限を安全に設定できます。

グループのリストは、**/etc/group** 設定ファイルに保存されます。

9.2. ユーザーアカウントの管理

Red Hat Enterprise Linux は、マルチユーザー向けのオペレーティングシステムです。つまり、1 台のマ

シンにインストールされた1つのシステムに、複数のユーザーが別々のコンピューターからアクセスできます。各ユーザーは自身のアカウントで操作します。このような方法でユーザーアカウントを管理することは、Red Hat Enterprise Linux のシステム管理の中心的要素になります。

以下に各種ユーザーアカウントを紹介します。

- **通常のユーザーアカウント**
  通常のアカウントは特定システムのユーザー用に作成されます。このようなアカウントは、通常のシステム管理中に追加、削除、および修正できます。

- **システムユーザーアカウント:**
  システムユーザーアカウントは、システムで特定のアプリケーション識別子を表します。このようなアカウントは通常、ソフトウェアのインストール時にのみ追加または操作され、後で変更することはありません。

警告

システムアカウントは、システムでローカルに利用できると想定されています。アカウントがリモートで設定され、提供されている (LDAP の設定など) と、システムが破損したり、サービスが開始できない場合があります。

  システムアカウント用に、1000 番未満のユーザー ID が予約されています。通常のアカウントには、1000 から始まる ID を使用できます。ただし、5000 以降の ID を割り当てることが推奨されます。ID の割り当ては、**/etc/login.defs** ファイルを参照してください。

- **グループ:**
  グループとは、複数のユーザーアカウントを共通目的 (特定のファイルにアクセス権を与えるなど) で統合するエンティティーです。

9.2.1. コマンドラインツールを使用したアカウントとグループの管理

ユーザーアカウントとグループを管理するには、次の基本的なコマンドラインツールを使用します。

- ユーザーおよびグループ ID を表示します。

```
$ 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
```

- 新規ユーザーアカウントを作成します。

```
# useradd example.user
```

- example.user に属するユーザーアカウントに新しいパスワードを割り当てます。

```
# passwd example.user
```

- ユーザーをグループに追加します。

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

関連情報

- **man useradd(8)**、**man passwd(1)**、および **man usermod(8)**

9.2.2. Web コンソールで管理されるシステムユーザーアカウント

RHEL Web コンソールに表示されているユーザーアカウントでは、以下が可能になります。

- システムにアクセスする際にユーザーを認証する
- システムへのアクセス権を設定する

RHEL Web コンソールは、システムに存在するすべてのユーザーアカウントを表示します。そのため、最初に Web コンソールにログインした直後は、ユーザーアカウントが少なくとも 1 つ表示されます。

RHEL Web コンソールにログインしたら、以下の操作を実行できます。

- 新規ユーザーアカウントの作成
- パラメーターの変更
- アカウントのロック
- ユーザーセッションの終了

9.2.3. Web コンソールで新規アカウントの追加

RHEL Web コンソールを使用して、システムにユーザーアカウントを追加し、アカウントに管理権限を設定できます。

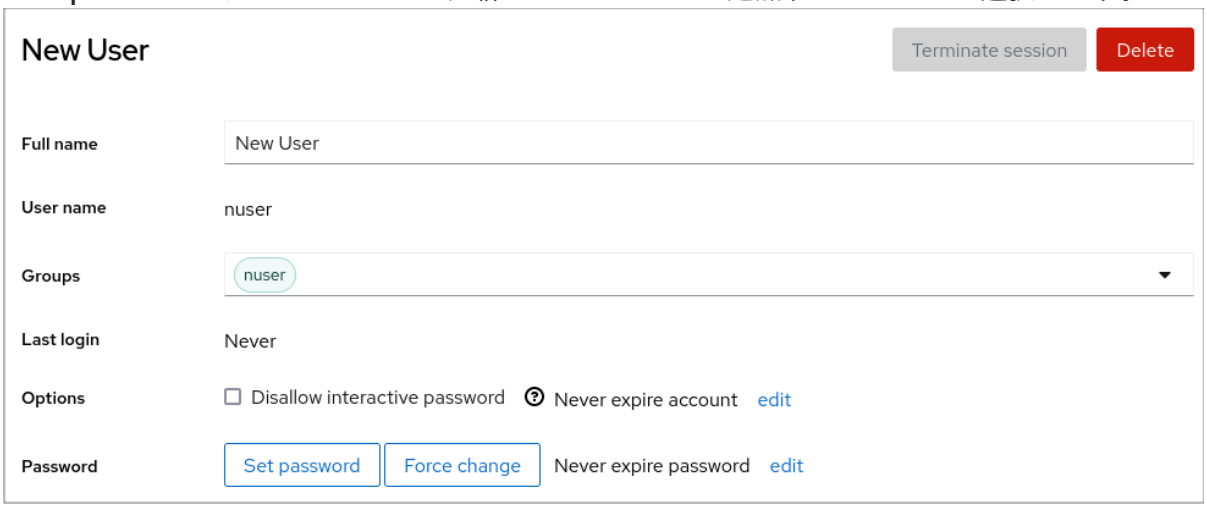
前提条件

- RHEL Web コンソールがインストールされており、アクセス可能である。詳細は、Web コンソールのインストール を参照してください。

手順

1. RHEL Web コンソールにログインします。
2. **Accounts** をクリックします。
3. **Create New Account** をクリックします。
4. **フルネーム** フィールドにユーザーの氏名を入力します。
   RHEL Web コンソールは、入力した氏名からユーザー名が自動的に作成され、**ユーザー名** フィールドに入力されます。名前の頭文字と、苗字で設定される命名規則を使用しない場合は、入力されたユーザー名を変更します。
5. **パスワード/確認** フィールドにパスワードを入力し、再度パスワードを入力します。
   フィールドの下にあるカラーバーは、入力したパスワードの強度を表し、弱いパスワードは使用できないようにします。
6. **作成** をクリックして設定を保存し、ダイアログボックスを閉じます。

7. 新規作成したアカウントを選択します。
8. **Groups** ドロップダウンメニューで、新しいアカウントに追加するグループを選択します。



これで **アカウント** 設定に新規アカウントが表示され、認証情報を使用してシステムに接続できるようになりました。

9.3. コマンドラインからのユーザーの管理

コマンドラインインターフェイス (CLI) を使用してユーザーおよびグループを管理できます。これにより、Red Hat Enterprise Linux 環境でユーザーおよびユーザーグループを追加、削除、および変更できます。

9.3.1. コマンドラインでの新規ユーザーの追加

useradd ユーティリティーを使用して、新規ユーザーを追加できます。

前提条件

- **Root** アクセス

手順

- 新規ユーザーを追加するには、以下を使用します。

```
# useradd options username
```

options は **useradd** コマンドのコマンドラインオプションに、username はユーザー名に置き換えます。

例9.1 新規ユーザーの追加

ユーザー ID が **5000** のユーザー **sarah** を追加するには、以下を使用します。

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

検証手順

- 新規ユーザーが追加されたことを確認するには、**id** ユーティリティーを使用します。

```
# id sarah
```

返される出力は以下のとおりです。

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

関連情報

- **useradd** の man ページ

9.3.2. コマンドラインでの新規グループの追加

groupadd ユーティリティーを使用して、新規グループを追加できます。

前提条件

- **Root** アクセス

手順

- 新規グループを追加するには、以下を使用します。

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

options は **groupadd** コマンドのコマンドラインオプションに、group-name はグループ名に置き換えます。

例9.2 新規グループの追加

グループ ID が **5000** のグループ **sysadmins** を追加するには、以下を使用します。

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

検証手順

- 新規グループが追加されていることを確認するには、**tail** ユーティリティーを使用します。

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

返される出力は以下のとおりです。

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

関連情報

- **groupadd** の man ページ

9.3.3. コマンドラインから補助グループにユーザーを追加

補助グループにユーザーを追加して、権限を管理したり、特定のファイルまたはデバイスへのアクセスを有効にしたりできます。

前提条件

- **root** アクセス

手順

- ユーザーの補助グループにグループを追加するには、以下を使用します。

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

group-name はグループ名に、username はユーザー名に置き換えます。

例9.3 補助グループへのユーザーの追加

ユーザーの **sysadmin** をグループ **system-administrators** に追加するには、以下を使用します。

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

検証手順

- ユーザー **sysadmin** の補助グループに新規グループが追加されていることを確認するには、以下を使用します。

```
# groups sysadmin
```

この出力では、以下が表示されます。

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

9.3.4. グループディレクトリーの作成

UPG システム設定では、**グループ ID 権限の設定** (setgid) をディレクトリーに適用できます。**setgid** を使用して、ディレクトリーを共有するグループプロジェクトの管理が簡単になります。**setgid** をディレクトリーに適用すると、ディレクトリー内に作成されたファイルは、そのディレクトリーを所有するグループに自動的に割り当てられます。このグループ内の書き込みおよび実行権限があるユーザーは、対象のディレクトリーにファイルを作成、変更、および削除できるようになりました。

次のセクションでは、グループディレクトリーを作成する方法を説明します。

前提条件

- **Root** アクセス

手順

1. ディレクトリーを作成します。

```
# mkdir directory-name
```

directory-name は、ディレクトリー名に置き換えます。

2. グループを作成します。

```
# groupadd group-name
```

group-name は、グループ名に置き換えます。

3. ユーザーをグループに追加します。

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

group-name はグループ名に、username はユーザー名に置き換えます。

4. ディレクトリーのユーザーとグループの所有権は、group-name グループに関連付けます。

```
# chgrp group-name directory-name
```

group-name はグループ名に、directory-name はディレクトリー名に置き換えます。

5. ファイルおよびディレクトリーを作成および修正し、**setgid** を設定してこの権限を directory-name ディレクトリー内で適用できるようにします。

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

directory-name は、ディレクトリー名に置き換えます。

group-name グループのすべてのメンバーが、**directory-name** ディレクトリーにファイルを作成し、編集できるようになりました。新規に作成されたファイルは、**group-name** グループの所有権を保持します。

検証手順

- パーミッション設定の正確性を検証するには、以下を使用します。

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

directory-name は、ディレクトリー名に置き換えます。

返される出力は以下のとおりです。

```
drwx**rws**r-x. 2 root **group-name** 6 Nov 25 08:45 **directory-name**
```

9.3.5. コマンドラインでのユーザーの削除

コマンドラインを使用してユーザーアカウントを削除できます。ユーザーアカウントの削除に加えて、必要に応じてユーザーデータとメタデータ (ホームディレクトリーや設定ファイルなど) を削除できます。

前提条件

- **root** アクセスがある。
- ユーザーが存在している。
- ユーザーがログアウトしていることを確認します。

```
# loginctl terminate-user user-name
```

手順

- ユーザーデータではなく、ユーザーアカウントのみを削除するには、以下を実行します。

```
# userdel user-name
```

- ユーザー、データ、およびメタデータを削除するには、以下を実行します。

a. ユーザー、そのホームディレクトリー、メールスプール、および SELinux ユーザーマッピングを削除します。

```
# userdel --remove --selinux-user user-name
```

b. 追加のユーザーメタデータを削除します。

```
# rm -rf /var/lib/AccountsService/users/user-name
```

このディレクトリーには、ホームディレクトリーが使用可能になる前にシステムが必要とする、ユーザーに関する情報が保存されます。システムの設定によっては、ユーザーがログイン画面で認証するまで、ホームディレクトリーが利用できない可能性があります。

重要

このディレクトリーを削除せずに、後で同じユーザーを再作成した場合、再作成されたユーザーは、削除されたユーザーから継承した特定の設定を引き続き使用します。

関連情報

- **userdel(8)** man ページ

9.4. WEB コンソールでユーザーアカウントの管理

RHEL Web コンソールは、直接ターミナルにアクセスせずに、幅広い管理タスクを実行できるグラフィカルインターフェイスを提供します。たとえば、システムユーザーアカウントの追加、編集、または削除が可能です。

本セクションの内容を読むと、以下を理解できます。

- 既存のアカウントが存在する場所
- 新規アカウントの追加方法
- パスワードの有効期限の設定方法
- ユーザーセッションを終了する方法および時期

前提条件

- RHEL Web コンソールを設定しておく。詳細はRHEL Web コンソールの使用ガイド を参照してください。
- 管理者権限が割り当てられたアカウントで RHEL Web コンソールにログインしている。詳細は RHEL Web コンソールへのログイン を参照してください。

9.4.1. Web コンソールで管理されるシステムユーザーアカウント

RHEL Web コンソールに表示されているユーザーアカウントでは、以下が可能になります。

- システムにアクセスする際にユーザーを認証する
- システムへのアクセス権を設定する

RHEL Web コンソールは、システムに存在するすべてのユーザーアカウントを表示します。そのため、最初に Web コンソールにログインした直後は、ユーザーアカウントが少なくとも 1 つ表示されます。

RHEL Web コンソールにログインしたら、以下の操作を実行できます。

- 新規ユーザーアカウントの作成
- パラメーターの変更
- アカウントのロック
- ユーザーセッションの終了

9.4.2. Web コンソールで新規アカウントの追加

RHEL Web コンソールを使用して、システムにユーザーアカウントを追加し、アカウントに管理権限を設定できます。

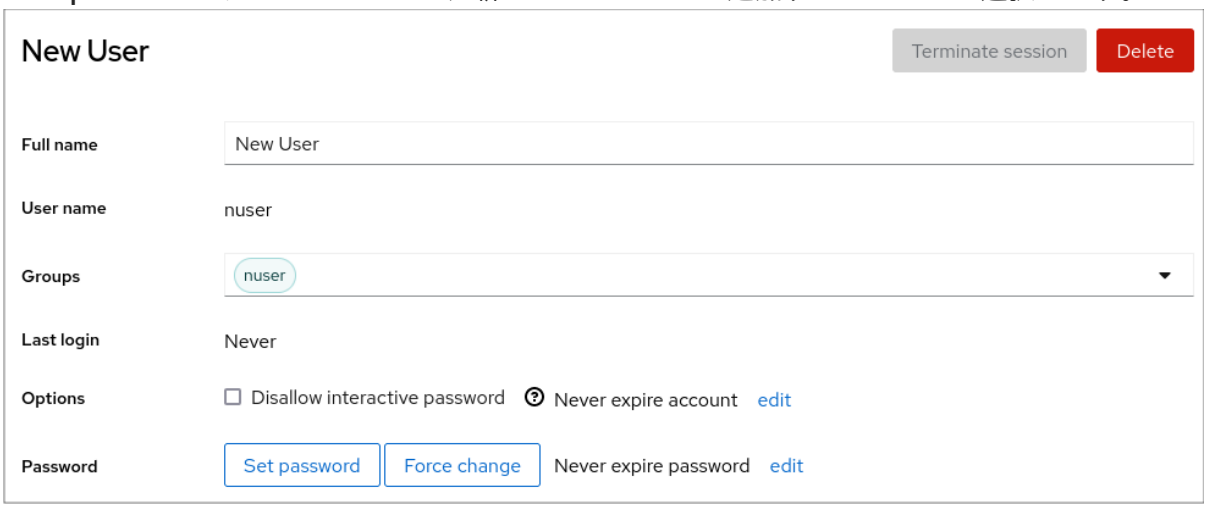
前提条件

- RHEL Web コンソールがインストールされており、アクセス可能である。詳細は、Web コンソールのインストール を参照してください。

手順

1. RHEL Web コンソールにログインします。
2. **Accounts** をクリックします。
3. **Create New Account** をクリックします。
4. **フルネーム** フィールドにユーザーの氏名を入力します。
   RHEL Web コンソールは、入力した氏名からユーザー名が自動的に作成され、**ユーザー名** フィールドに入力されます。名前の頭文字と、苗字で設定される命名規則を使用しない場合は、入力されたユーザー名を変更します。
5. **パスワード/確認** フィールドにパスワードを入力し、再度パスワードを入力します。
   フィールドの下にあるカラーバーは、入力したパスワードの強度を表し、弱いパスワードは使用できないようにします。

6. **作成** をクリックして設定を保存し、ダイアログボックスを閉じます。
7. 新規作成したアカウントを選択します。
8. **Groups** ドロップダウンメニューで、新しいアカウントに追加するグループを選択します。



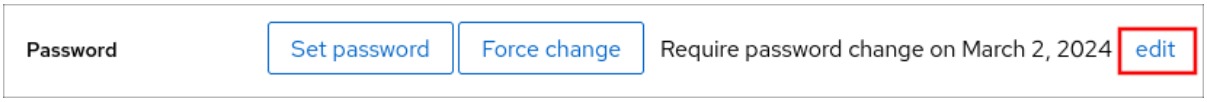
これで **アカウント** 設定に新規アカウントが表示され、認証情報を使用してシステムに接続できるようになりました。

9.4.3. Web コンソールでパスワード有効期限の強制

デフォルトでは、ユーザーアカウントのパスワードに期限はありません。定義した日数が経過したら、システムパスワードが期限切れになるように設定できます。パスワードが期限切れになると、次回のログイン時にパスワードの変更が要求されます。

手順

1. RHEL 9 Web コンソールにログインします。
2. **Accounts** をクリックします。
3. パスワードの有効期限を設定するユーザーアカウントを選択します。
4. **Password** 行の **edit** をクリックします。



5. **Password expiration** ダイアログボックスで、**Require password change every ... days**を選択し、パスワードの期限が切れる日数を示した、正の整数を入力します。
6. **Change** をクリックします。
Web コンソールの **Password** 行に、将来のパスワード変更リクエストの日付がすぐに表示されます。

9.4.4. Web コンソールでユーザーセッションの終了

ユーザーがシステムにログインすると、ユーザーセッションが作成されます。ユーザーセッションを終了すると、ユーザーはシステムからログアウトされます。これは、システムのアップグレードなどの、設定変更の影響を受ける管理タスクを実行する必要がある場合に便利です。

RHEL 9 Web コンソールの各ユーザーアカウントで、現在使用している Web コンソールセッション以外のセッションすべてを終了できます。これにより、システムへの不正アクセスを阻止できます。

手順

1. RHEL 9 Web コンソールにログインします。
2. **Accounts** をクリックします。
3. セッションを終了するユーザーアカウントをクリックします。
4. **Terminate Session** をクリックします。
   Terminate Session ボタンが無効になっている場合は、ユーザーがシステムにログインしていません。

   RHEL Web コンソールはセッションを終了します。

9.5. コマンドラインを使用したユーザーグループの編集

ユーザーは、ファイルおよびフォルダーに同様のアクセスを持つユーザーの論理的な集合を許可する、特定のグループセットに属します。コマンドラインから、プライマリーユーザーグループおよび補助ユーザーグループを編集して、ユーザーの権限を変更できます。

9.5.1. プライマリーユーザーグループおよび補助ユーザーグループ

グループとは、複数のユーザーアカウントを共通目的 (特定のファイルにアクセス権を与えるなど) で統合するエンティティーです。

Linux では、ユーザーグループはプライマリーまたは補助として機能できます。プライマリーグループおよび補助グループには、以下のプロパティーがあります。

プライマリーグループ

- すべてのユーザーに、常に 1 つのプライマリーグループのみが存在します。
- ユーザーのプライマリーグループは変更できます。

補助グループ

- 既存の補助グループに既存のユーザーを追加して、グループ内で同じセキュリティーおよびアクセス権限を持つユーザーを管理できます。
- ユーザーは、ゼロまたは複数の補助グループのメンバーになります。

9.5.2. ユーザーのプライマリーグループおよび補助グループのリスト表示

ユーザーのグループをリスト表示して、どのプライマリーグループおよび補助グループに属しているかを確認できます。

手順

- ユーザーのプライマリーおよび補助グループの名前を表示します。

```
$ groups user-name
```

user-name は、ユーザー名に置き換えます。ユーザー名を指定しないと、コマンドは現在のユーザーのグループメンバーシップを表示します。最初のグループはプライマリーグループで、その後に任意の補助グループが続きます。

例9.4 ユーザー sarah のグループのリスト表示

```
$ groups sarah
```

この出力では、以下が表示されます。

```
sarah : sarah wheel developer
```

ユーザー **sarah** にはプライマリーグループ **sarah** があり、補助グループ **wheel** および **developer** のメンバーになります。

例9.5 ユーザー marc のグループのリスト表示

```
$ groups marc
```

この出力では、以下が表示されます。

```
marc : marc
```

ユーザー **marc** には、プライマリーグループ **marc** のみがあり、補助グループはありません。

9.5.3. ユーザーのプライマリーグループの変更

既存ユーザーのプライマリーグループを、新しいグループに変更できます。

前提条件:

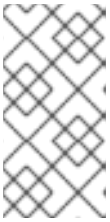
1. **root** アクセス
2. 新しいグループが存在する必要があります。

手順

- ユーザーのプライマリーグループを変更します。

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

group-name を、新しいプライマリーグループの名前に置き換え、**user-name** を、ユーザーの名前に置き換えます。



注記

ユーザーのプライマリーグループを変更すると、コマンドは、ユーザーのホームディレクトリーにあるすべてのファイルのグループ所有権も、自動的に新しいプライマリーグループに変更します。ユーザーのホームディレクトリー外のファイルのグループ所有権を手動で修正する必要があります。

例9.6 ユーザーのプライマリーグループを変更する例:

ユーザー **sarah** がプライマリーグループ **sarah1** に所属しており、ユーザーのプライマリーグループを **sarah2** に変更する場合は、以下を使用します。

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

検証手順

- ユーザーのプライマリーグループを変更したことを確認します。

```
$ groups sarah
```

この出力では、以下が表示されます。

```
sarah : sarah2
```

9.5.4. コマンドラインから補助グループにユーザーを追加

補助グループにユーザーを追加して、権限を管理したり、特定のファイルまたはデバイスへのアクセスを有効にしたりできます。

前提条件

- **root** アクセス

手順

- ユーザーの補助グループにグループを追加するには、以下を使用します。

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

group-name はグループ名に、username はユーザー名に置き換えます。

例9.7 補助グループへのユーザーの追加

ユーザーの **sysadmin** をグループ **system-administrators** に追加するには、以下を使用します。

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

検証手順

- ユーザー **sysadmin** の補助グループに新規グループが追加されていることを確認するには、以下を使用します。

```
# groups sysadmin
```

この出力では、以下が表示されます。

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

9.5.5. 補助グループからユーザーの削除

補助グループから既存のユーザーを削除して、権限や、ファイルやデバイスへのアクセスを制限できます。

前提条件

- **root** アクセス

手順

- 補助グループからユーザーを削除します。

```
# gpasswd -d user-name group-name
```

user-name をユーザー名に置き換え、group-name を、補助グループの名前に置き換えます。

例9.8 補助グループからユーザーの削除

ユーザー sarah にプライマリーグループ **sarah2** があり、セカンダリーグループ **wheel** および **developers** に属し、そのユーザーをグループ **developers** から削除する場合は、次のコマンドを実行します。

```
# gpasswd -d sarah developers
```

検証手順

- セカンダリーグループの開発者からユーザー sarah を削除したことを確認します。

```
$ groups sarah
```

この出力では、以下が表示されます。

```
sarah : sarah2 wheel
```

9.5.6. ユーザーの補助グループのすべての変更

ユーザーをメンバーとして残す補助グループのリストを上書きできます。

前提条件

- **root** アクセス
- 補助グループが存在している

手順

- ユーザーの補助グループのリストを上書きします。

```
# usermod -G group-names username
```

group-names を、1 つ以上の補助グループの名前に置き換えます。ユーザーを複数の補助グループに一度に追加するには、グループ名をコンマで区切り、スペースを使用しないでください。たとえば、**wheel,developer** です。

user-name は、ユーザー名に置き換えます。

重要

ユーザーが、指定しないグループのメンバーである場合は、グループからそのユーザーが削除されます。

例9.9 ユーザーの補助グループのリストの変更

ユーザー **sarah** にプライマリーグループ **sarah2** があり、補助グループ **wheel** に属し、さらに 3 つの補助グループ **developer**、**sysadmin**、および **security** に属するユーザーにする場合は、次のコマンドを実行します。

```
# usermod -G wheel,developer,sysadmin,security sarah
```

検証手順

- 補助グループのリストが正しく設定されていることを確認します。

```
# groups sarah
```

この出力では、以下が表示されます。

```
sarah : sarah2 wheel developer sysadmin security
```

9.6. ROOT パスワードの変更およびリセット

既存の root パスワードが要件を満たさないか、パスワードを忘れた場合には、**root** ユーザーおよび root 以外のユーザーの両方として、パスワードを変更またはリセットできます。

9.6.1. root ユーザーとしての root パスワードの変更

passwd コマンドを使用して、**root** ユーザーとして **root** パスワードを変更できます。

前提条件

- **Root** アクセス

手順

- **root** パスワードを変更する場合は、次のコマンドを実行します。

```
# passwd
```

変更する前に、現在のパスワードを入力するように求められます。

9.6.2. root 以外のユーザーが root パスワードを変更またはリセット

passwd コマンドを使用して、root 以外のユーザーとして **root** パスワードを変更またはリセットできます。

前提条件

- root 以外のユーザーとしてログインできる。
- **wheel** 管理グループのメンバーである。

手順

- **wheel** グループに属する root ユーザー以外のユーザーとして **root** パスワードを変更またはリセットするには、以下を使用します。

```
$ sudo passwd root
```

root パスワードを変更する前に、root 以外のユーザーのパスワードを入力するように求められます。

9.6.3. 起動時の root パスワードのリセット

root 以外のユーザーとしてログインできない場合や、**wheel** 管理グループに所属しない場合は、特別な **chroot jail** 環境に切り替えることで起動時に root パスワードをリセットできます。

手順

1. システムを再起動して、GRUB 2 ブート画面で **e** キーを押して、起動プロセスを中断します。カーネルブートパラメーターが表示されます。

```
load_video
set gfx_payload=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. **linux** で始まる行の最後に移動します。

```
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
```

Ctrl+e を押して、行の最後に移動します。

3. **rd.break** を **linux** で始まる行の最後に追加します。

```
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. **Ctrl+x** を押して、変更したパラメーターでシステムを起動します。
switch_root プロンプトが表示されます。

5. ファイルシステムを書き込み可能で再マウントします。

```
mount -o remount,rw /sysroot
```

ファイルシステムは、**/sysroot** ディレクトリーに読み取り専用としてマウントされます。書き込み可能なファイルシステムとして再マウントすると、パスワードを変更できます。

6. **chroot** 環境に入ります。

```
chroot /sysroot
```

sh-4.4# プロンプトが表示されます。

7. **root** パスワードをリセットします。

```
passwd
```

コマンドラインに表示される指示に従って、**root** パスワードの変更を完了します。

8. 次回のシステム起動時に SELinux の再ラベルプロセスを有効にします。

```
touch /.autorelabel
```

9. **chroot** 環境を終了します。

```
exit
```

10. **switch_root** プロンプトを終了します。

```
exit
```

11. SELinux の再ラベルプロセスが終了するまで待機します。大規模なディスクの再ラベルには時間がかかる可能性があることに注意してください。プロセスが完了すると、システムが自動的に再起動します。

検証手順

1. **root** パスワードが正常に変更されたことを確認するには、通常のユーザーとしてログインし、ターミナルを開きます。

2. root としてインタラクティブシェルを実行します。

```
$ su
```

3. 新しい **root** パスワードを入力します。

4. 現在の実効ユーザー ID に関連付けられたユーザー名を出力します。

```
# whoami
```

返される出力は以下のとおりです。

```
root
```

第10章 SUDO アクセスの管理

システム管理者は、root 以外のユーザーに、通常 root ユーザー用に予約されている管理コマンドを実行できるようにする **sudo** アクセスを付与できます。これにより、root 以外のユーザーは、root ユーザーアカウントにログインせずに、このようなコマンドを実行できます。

10.1. SUDOERS のユーザー認可

/etc/sudoers ファイルは、どのユーザーが **sudo** コマンドを使用して他のコマンドを実行できるかを指定します。ルールは、個別のユーザーおよびユーザーグループに適用できます。エイリアスを使用して、ホスト、コマンド、さらにはユーザーのグループに対するルールをより簡単に定義することもできます。デフォルトのエイリアスは、**/etc/sudoers** ファイルの最初の部分で定義されます。

認可されていないユーザーが **sudo** を使用してコマンドを入力すると、システムは文字列 **<username> : user NOT in sudoers** を含むメッセージをジャーナルログに記録します。

デフォルトの **/etc/sudoers** ファイルは、認可の情報と例を提供します。対応する行をコメント解除すると、特定のルール例をアクティブにできます。ユーザー認可に関するセクションには、以下の概要が示されます。

```
## Next comes the main part: which users can run what software on
## which machines  (the sudoers file can be shared between multiple
## systems).
```

次の形式を使用して、新しい **sudoers** 認可を作成したり、既存の認可を変更したりできます。

```
<username> <hostname.example.com>=(<run_as_user>:<run_as_group>) <path/to/command>
```

ここでは、以下のようになります。

- **<username>** はコマンドを入力するユーザーです (例: **user1**)。値が **%** で始まる場合はグループを定義します (例: **%group1**)。
- **<hostname.example.com>** は、ルールが適用されるホストの名前です。
- セクション **(<run_as_user>:<run_as_group>)** は、コマンドを実行するユーザーまたはグループを定義します。このセクションを省略すると、**<username>** は root としてコマンドを実行できます。
- **<path/to/command>** は、コマンドへの完全な絶対パスです。また、コマンドパスの後にオプションを追加することにより、特定のオプションおよび引数を指定したコマンドのみを実行するようにユーザーを制限することもできます。オプションを指定しないと、すべてのオプションが有効な状態でコマンドを使用できます。

これらの変数のいずれかを **ALL** に置き換えることで、ルールをすべてのユーザー、ホスト、またはコマンドに適用できます。

警告

ALL ALL=(ALL) ALL などの過度に寛容なルールを使用すると、すべてのユーザーがすべてのコマンドを、いずれのホストのいずれのユーザーとしても使用できます。これは重大なセキュリティーリスクをもたらします。

! 演算子を使用して引数を負の値で指定できます。たとえば、**!root** は root 以外のすべてのユーザーを指定します。特定のユーザー、グループ、およびコマンドを許可する方が、特定のユーザー、グループ、およびコマンドを拒否するよりも安全です。これは、許可ルールにより、認可されていない新規ユーザーまたはグループもブロックされるためです。

警告

alias コマンドでコマンドの名前を変更すると、このような規則が上書きされるため、コマンドに否定的な規則を使用しないでください。

システムは、**/etc/sudoers** ファイルを最初から最後まで読み取ります。したがって、ファイルにユーザーの複数のエントリーが含まれている場合、エントリーは順番に適用されます。値が競合する場合は、最も具体的な一致ではない場合でも、システムは最後の一致を使用します。

システム更新中にルールを保持し、エラーを簡単に修正するには、ルールを **/etc/sudoers** ファイルに直接入力するのではなく、**/etc/sudoers.d/** ディレクトリーに新しいファイルを作成して、新しいルールを入力します。システムは、**/etc/sudoers** ファイル内で以下の行に到達する際に、**/etc/sudoers.d** ディレクトリー内のファイルを読み取ります。

```
#includedir /etc/sudoers.d
```

この行の頭にある番号記号 (**#**) は構文の一部であり、行がコメントであることを意味するものではありません。そのディレクトリー内のファイル名にはピリオドを使用することができません。また、末尾にチルダ (~) を使用することもできません。

関連情報

- **sudo(8)** および **sudoers(5)** の man ページ

10.2. ユーザーへの SUDO アクセス権限の付与

システム管理者は、root 以外のユーザーに **sudo** アクセスを付与することで、管理コマンドの実行を許可できます。**sudo** コマンドは、root ユーザーのパスワードを使用せずに、管理アクセスを持つユーザーを提供する方法です。

ユーザーが管理コマンドを実行する必要がある場合には、コマンドの前に **sudo** を指定してください。ユーザーがコマンドに対する認可を持っている場合、コマンドは root であるかのように実行されます。

以下の制限事項に注意してください。

- **/etc/sudoers** 設定ファイルにリスト表示されているユーザーのみが **sudo** コマンドを使用できます。
- コマンドは、root シェルではなく、ユーザーのシェルで実行されます。

前提条件

- システムへの root アクセス権があります。

手順

1. root で **/etc/sudoers** ファイルを開きます。

```
# visudo
```

   /etc/sudoers ファイルは、**sudo** コマンドで適用されるポリシーを定義します。

2. **/etc/sudoers** ファイルで、**wheel** 管理グループのユーザーに **sudo** アクセスを付与する行を見つけます。

```
## Allows people in group wheel to run all commands
%wheel        ALL=(ALL)       ALL
```

3. **%wheel** で始まる行が番号記号 (**#**) でコメントアウトされていないことを確認してください。
4. 変更を保存し、エディターを終了します。
5. **sudo** アクセスを付与するユーザーを、**wheel** 管理グループに追加します。

```
 # usermod --append -G wheel <username>
```

   <username> は、ユーザー名に置き換えます。

検証手順

- ユーザーが **wheel** 管理グループに含まれていることを確認します。

```
# id <username>
uid=5000(<username>) gid=5000(<username>) groups=5000(<username>),10(wheel)
```

関連情報

- **sudo(8)**、**visudo(8)**、および **sudoers(5)** の man ページ

10.3. 非特権ユーザーが特定のコマンドを実行できるようにする

管理者は、**/etc/sudoers.d/** ディレクトリーにポリシーを設定することで、非特権ユーザーが特定のワークステーションに特定のコマンドを入力できるようにすることができます。

前提条件

- システムへの root アクセス権があります。

手順

1. root で、**/etc/** の下に新しい **sudoers.d** ディレクトリーを作成します。

```
# mkdir -p /etc/sudoers.d/
```

2. **/etc/sudoers.d** ディレクトリーに新規ファイルを作成します。

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

ファイルが自動的に開きます。

3. 次の行を **/etc/sudoers.d/<filename>** ファイルに追加します。

```
<username> <hostname.example.com> = (<run_as_user>:<run_as_group>)
<path/to/command>
```

- **<username>** は、ユーザー名に置き換えます。
- **<hostname.example.com>** は、ホストの URL に置き換えます。
- **(<run_as_user>:<run_as_group>)** は、コマンドを実行できるユーザーまたはグループに置き換えます。このセクションを省略すると、**<username>** は root としてコマンドを実行できます。
- **<path/to/command>**は、コマンドへの完全な絶対パスに置き換えます。また、コマンドパスの後にオプションを追加することにより、特定のオプションおよび引数を指定したコマンドのみを実行するようにユーザーを制限することもできます。オプションを指定しないと、すべてのオプションが有効な状態でコマンドを使用できます。
- 同じホストで 2 つ以上のコマンドを 1 行で許可するには、コンマで区切り、コンマの後にスペースを付けることができます。
  たとえば、**user1** が **host1.example.com** で **dnf** および **reboot** コマンドを実行できるようにするには、**user1 host1.example.com = /bin/dnf, /sbin/reboot** と入力します。

4. オプション: ユーザーが **sudo** 特権の使用を試みるたびにメール通知を受け取るには、以下の行をファイルに追加します。

```
Defaults    mail_always
Defaults    mailto="<email@example.com>"
```

5. 変更を保存し、エディターを終了します。

検証

1. ユーザーが **sudo** 特権でコマンドを実行できるかどうかを確認するには、アカウントを切り替えます。

```
# su <username> -
```

2. ユーザーとして、**sudo** コマンドを使用してコマンドを入力します。

```
$ sudo <command>
[sudo] password for <username>:
```

ユーザーの **sudo** パスワードを入力します。

3. 特権が正しく設定されている場合、システムはコマンドとオプションのリストを表示します。たとえば、**dnf** コマンドを使用すると、次の出力が表示されます。

```
...
usage: dnf [options] COMMAND
...
```

システムが **<username> is not in the sudoers file. This incident will be reported** というエラーメッセージを返した場合、**/etc/sudoers.d/** に **<username>** のファイルが存在しません。

システムが **<username> is not allowed to run sudo on <host.example.com>** というエラーメッセージを返した場合、設定が正しく完了していません。root としてログインしていること、および設定が正しく実行されていることを確認してください。

システムが **Sorry, user <username> is not allowed to execute '<path/to/command>' as root on <host.example.com>.** というエラーメッセージを返した場合、コマンドがユーザーのルールで正しく定義されていません。

関連情報

- **sudo(8)**、**visudo(8)**、および **sudoers(5)** の man ページ

第11章 ファイルシステムの権限の管理

ファイルシステムの権限は、ユーザーおよびグループアカウントがファイルの内容を読み取り、変更し、実行する権限、およびディレクトリーに入る権限を制御します。権限を慎重に設定して、不正アクセスからデータを保護します。

11.1. ファイル権限の管理

すべてのファイルまたはディレクトリーには、以下のレベルの所有権があります。

- ユーザー所有者 (**u**)。
- グループの所有者 (**g**)。
- その他 (**o**)。

各所有権レベルには、以下のパーミッションを割り当てることができます。

- 読み取り (**r**)。
- 書き込み (**w**)。
- 実行 (**x**)。

ファイルの execute 権限があると、そのファイルを実行するできることに注意してください。ディレクトリーの実行権限では、ディレクトリーのコンテンツにアクセスできますが、実行はできません。

新規ファイルまたはディレクトリーが作成されると、デフォルトのパーミッションセットが自動的に割り当てられます。ファイルまたはディレクトリーのデフォルトパーミッションは、以下の 2 つの要素に基づいています。

- 基本パーミッション。
- **ユーザーのファイル作成モードマスク** (umask)

11.1.1. ベースファイルのパーミッション

新規ファイルまたはディレクトリーが作成されるたびに、基本パーミッションが自動的に割り当てられます。ファイルまたはディレクトリーの基本パーミッションは、**シンボリック** または **8 進数** の値で表すことができます。

パーミッション	**シンボリック値**	**8 進数値**
パーミッションなし	---	0
実行	--x	1
書き込み	-w-	2
書き込みおよび実行	-wx	3
読み取り	r--	4

読み取りおよび実行	r-x	5
読み取りおよび書き込み	rw-	6
読み取り、書き込み、実行	rwx	7

ディレクトリーの基本パーミッションは **777** (**drwxrwxrwx**) で、すべてのユーザーに読み取り、書き込み、実行権限を付与します。つまり，ディレクトリーの所有者、グループ、その他のユーザーはディレクトリーのコンテンツ表示、そのディレクトリーでの項目の作成、削除、編集、サブディレクトリーへの移動が可能です。

ディレクトリー内の個別ファイルには、ディレクトリーに無制限にアクセスできるにも拘らず、編集ができない独自のパーミッションが割り当てられている可能性があります。

ファイルの基本パーミッションは **666** (**-rw-rw-rw-**) で、すべてのユーザーに読み取りおよび書き込み権限を付与します。ファイルの所有者、グループ、その他のユーザーは、ファイルの読み取りと編集が可能です。

例11.1 ファイルのパーミッション

ファイルに以下のパーミッションがある場合:

```
$ ls -l
-rwxrw----. 1 sysadmins sysadmins 2 Mar 2 08:43 file
```

- **-** ファイルであることを示します。
- **rwx** は、ファイルの所有者にファイルの読み取り、書き込み、実行のパーミッションがあることを示します。
- **rw-** は、グループに読み取りと書き込みのパーミッションがあるが、ファイルの実行はできません。
- **---** は、他のユーザーにファイルの読み取り、書き込み、実行のパーミッションがないことを示します。
- **.** は、SELinux セキュリティーコンテキストがファイルに設定されていることを示しています。

例11.2 ディレクトリーのパーミッション

ディレクトリーに以下のパーミッションがある場合:

```
$ ls -dl directory
drwxr-----. 1 sysadmins sysadmins 2 Mar 2 08:43 directory
```

- **d** は、ディレクトリーであることを示します。
- **rwx** は、ディレクトリーの所有者にディレクトリーの内容を読み取り、書き込み、およびアクセスするパーミッションがあることを示します。
  ディレクトリーの所有者は、ディレクトリー内のアイテム (ファイル、サブディレクトリー) を表示して、アイテムのコンテンツへのアクセスや変更が可能です。

- **r-x** は、そのグループがディレクトリーの内容を読み取るパーミッションを持っているが、書き込み (新しいエントリーの作成やファイルの削除) はできないことを示します。**x** パーミッションは、**cd** コマンドを使用してディレクトリーにアクセスできることを意味します。
- **---** は、他のユーザーがディレクトリーの内容の読み取り、書き込み、またはアクセスパーミッションがないことを示します。
  ディレクトリーのユーザー所有者またはグループ所有者ではない場合に、そのディレクトリーのアイテムを表示したり、アイテムの情報にアクセスしたり、変更したりできません。
- **.** は、SELinux セキュリティーコンテキストがディレクトリーに設定されていることを示しています。

注記

ファイルまたはディレクトリーに自動的に割り当てられる基本パーミッションは、最終的にファイルまたはディレクトリーに指定されるデフォルトのパーミッション **ではありません**。ファイルまたはディレクトリーを作成すると、**umask** により基本パーミッションが変更されます。基本パーミッションと **umask** の組み合わせにより、ファイルおよびディレクトリーのデフォルトパーミッションが作成されます。

11.1.2. ユーザーのファイル作成モードマスク

ユーザーファイル作成モードマスク (**umask**) は、新規作成ファイルおよびディレクトリーにファイル権限を設定する方法を制御する変数です。**umask** は、基本パーミッション値からパーミッションを自動的に削除して、Linux システムの全体的なセキュリティーを強化します。**umask** は、**シンボリック** 値または **8 進数** の値で表すことができます。

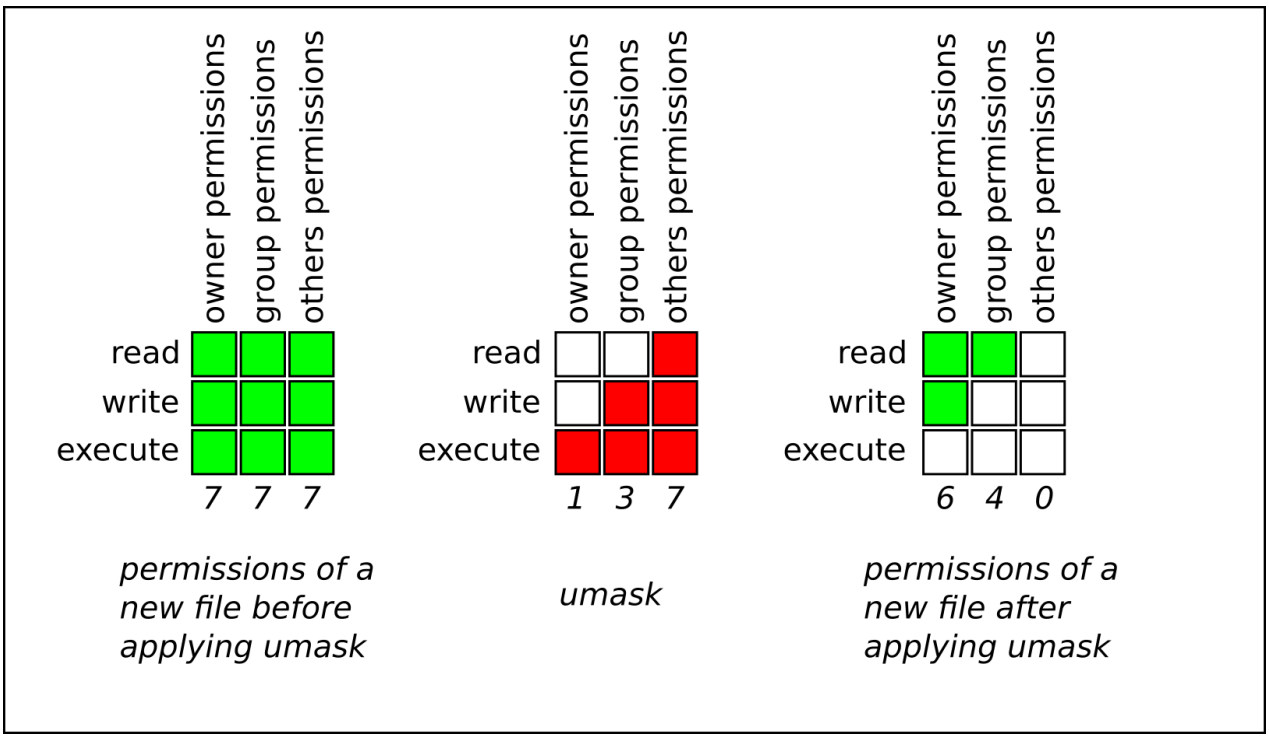
パーミッション	**シンボリック値**	**8 進数値**
読み取り、書き込み、実行	rwx	0
読み取りおよび書き込み	rw-	1
読み取りおよび実行	r-x	2
読み取り	r--	3
書き込みおよび実行	-wx	4
書き込み	-w-	5
実行	--x	6
権限なし	---	7

標準ユーザーと **root** ユーザーの両方のデフォルトの **umask** は **0022** です。

umask の最初の数字は、特別なパーミッション (スティッキービット) を表します。umask の最後の 3 桁はユーザーの所有者 (u)、グループの所有者 (g) およびその他 (o) からそれぞれ削除したパーミッションを表します。

例11.3 ファイルの作成時に umask を適用

以下の例では、umask の 8 進数値 **0137** が基本パーミッション **777** に適用され、デフォルトパーミッション **640** のファイルが作成されます。



11.1.3. デフォルトのファイル権限

デフォルトのパーミッションは、新規作成ファイルおよびディレクトリーに対して自動的に設定されます。デフォルトのパーミッションの値は、umask を基本パーミッションに適用して決定します。

例11.4 ディレクトリーのデフォルトの権限

標準ユーザー または root **ユーザー** が新しい **ディレクトリー** を作成すると、umask は **022** (**rwxr-xr-x**) に設定され、ディレクトリーの基本権限は **777** (**rwxrwxrwx**) に設定されます。これにより、デフォルトのパーミッションが **755** (**rwxr-xr-x**) になります。

	シンボリック値	**8 進数値**
基本パーミッション	rwxrwxrwx	777
Umask	rwxr-xr-x	022
デフォルトパーミッション	rwxr-xr-x	755

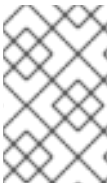
このパーミッションでは、ディレクトリーの所有者はディレクトリーの内容の表示、ディレクトリー内のアイテムの作成、削除、編集、サブディレクトリーへの移動が可能です。グループとその他のユーザーは、ディレクトリーの内容の表示とサブディレクトリーの移動のみが可能です。

例11.5 ファイルのデフォルトの権限

標準ユーザー または **root ユーザー** が新しい **ファイル** を作成すると、umask は **022** (**rwxr-xr-x**) に設定され、ファイルの基本権限は **666** (**rw-rw-rw-**) に設定されます。これにより、デフォルトのパーミッションが **644** (**-rw-r—r--**) になります。

	シンボリック値	**8 進数値**
基本パーミッション	rw-rw-rw-	666
Umask	rwxr-xr-x	022
デフォルトパーミッション	rw-r—r--	644

このパーミッションでは、ファイルの所有者はファイルを読み取りと編集が可能ですが、グループや他のユーザーはこのファイルの読み取りのみが可能です。



注記

セキュリティー上の理由から、通常のファイルに umask が **000** (**rwxrwxrwx**) に設定されていても、デフォルトで実行権限がありません。ただし、ディレクトリーは実行権限を持つ状態で作成できます。

11.1.4. シンボリック値を使用したファイル権限の変更

chmod ユーティリティーにシンボリック値 (組み合わせ文字および記号) を付けて、ファイルまたはディレクトリーのファイルのパーミッションを変更できます。

以下の **パーミッション** を割り当てることができます。

- 読み取り (**r**)
- 書き込み (**w**)
- 実行 (**x**)

パーミッションは、以下の **レベルの所有権** に割り当てることができます。

- ユーザー所有者 (**u**)
- グループ所有者 (**g**)
- その他 (**o**)
- すべて (**a**)

パーミッションを追加または削除するには、以下の **記号** を使用できます。

- **+**: 既存のパーミッションの上にパーミッションを追加します。
- **-**: 既存のパーミッションからパーミッションを削除します。
- **=**: 既存のパーミッションを削除し、新しいパーミッションを明示的に定義します。

手順

- ファイルまたはディレクトリーのパーミッションを変更するには、以下を使用します。

```
$ chmod <level><operation><permission> file-name
```

<level> は、パーミッションを設定する 所有権のレベル に置き換えます。**<operation>** は、署名 の 1 つに置き換えます。**<permission>** は、割り当てる パーミッション に置き換えます。**file-name** は、ファイルまたはディレクトリーの名前に置き換えます。たとえば、すべてのユーザーに読み取り、書き込み、実行 (**rwx**) **my-script.sh** のパーミッションを付与するには、**chmod a=rwx my-script.sh** コマンドを使用します。

詳細は ベースファイルのパーミッション を参照してください。

検証手順

- 特定のファイルのパーミッションを表示するには、以下を使用します。

```
$ ls -l file-name
```

file-name は、ファイルの名前に置き換えます。

- 特定のディレクトリーのパーミッションを表示するには、以下を使用します。

```
$ ls -dl directory-name
```

directory-name は、ディレクトリー名に置き換えます。

- 特定のディレクトリー内の全ファイルのパーミッションを表示するには、以下を使用します。

```
$ ls -l directory-name
```

directory-name は、ディレクトリー名に置き換えます。

例11.6 ファイルおよびディレクトリーの権限の変更

- **my-file.txt** のパーミッションを **-rw-rw-r--** から **-rw------** に変更するには以下を使用します。

1. **my-file.txt** の現在のパーミッションを表示します。

```
$ ls -l my-file.txt
-rw-rw-r--. 1 username username 0 Feb 24 17:56 my-file.txt
```

2. グループ所有者 (g) およびその他のファイル (**o**) からファイルを読み取り、書き込み、**実行** (**rwx**) するパーミッションを削除します。

```
$ chmod go= my-file.txt
```

パーミッションを等号 (**=**) の後ろに指定していない場合には自動的に無視される点に注意してください。

3. **my-file.txt** のパーミッションが正しく設定されていることを確認します。

```
$ ls -l my-file.txt
-rw-------. 1 username username 0 Feb 24 17:56 my-file.txt
```

- **my-directory** のパーミッションを **drwxrwx---** から **drwxrwxr-x** に変更するには、以下を使用します。

1. **my-directory** の現在のパーミッションを表示します。

```
$ ls -dl my-directory
drwxrwx---. 2 username username 4096 Feb 24 18:12 my-directory
```

2. すべてのユーザー (**a**) に対する読み取りおよび実行 (**r-x**) アクセスを追加します。

```
$ chmod o+rx my-directory
```

3. **my-directory** とそのコンテンツが正しく設定されていることを確認します。

```
$ ls -dl my-directory
drwxrwxr-x. 2 username username 4096 Feb 24 18:12 my-directory
```

11.1.5. 8 進数値を使用したファイル権限の変更

chmod ユーティリティーに 8 進数値 (数値) を指定して chmod ユーティリティーを使用し、ファイルまたはディレクトリーのファイルパーミッションを変更できます。

手順

- 既存のファイルまたはディレクトリーのファイル権限を変更するには、以下を使用します。

```
$ chmod octal_value file-name
```

file-name は、ファイルまたはディレクトリーの名前に置き換えます。**octal_value** は 8 進数値に置き換えます。詳細は ベースファイルのパーミッション を参照してください。

11.2. アクセス制御リストの管理

各ファイルおよびディレクトリーには、ユーザー所有者とグループ所有者を一度に指定できます。他のファイルやディレクトリーを非公開のままにし、別のユーザーまたはグループが所有する特定のファイルまたはディレクトリーにアクセスできるようなユーザーのパーミッションを付与する場合には、Linux アクセス制御リスト (ACL) を使用できます。

11.2.1. 現在のアクセス制御リストの表示

getfacl ユーティリティーを使用して、現在の ACL を表示できます。

手順

- 特定のファイルまたはディレクトリーの現在の ACL を表示するには、以下を使用します。

```
$ getfacl file-name
```

file-name は、ファイルまたはディレクトリーの名前に置き換えます。

11.2.2. アクセス制御リストの設定

setfacl ユーティリティーを使用して、ファイルまたはディレクトリーに ACL を設定できます。

前提条件

- **root** アクセス

手順

- ファイルまたはディレクトリーに ACL を設定するには、以下を使用します。

```
# setfacl -m u:username:symbolic_value file-name
```

username はユーザー名に、symbolic_value はシンボリック値に、file-name はファイルまたはディレクトリーの名前に置き換えます。詳細は、 **setfacl** の man ページを参照してください。

例11.7 グループプロジェクトのパーミッションの変更

以下の例では、**root** グループに所属する **root** ユーザーが所有する **group-project** ファイルのパーミッションを修正する方法を説明します。このファイルは以下のように設定します。

- 誰にも実行権限がない。
- ユーザー **andrew** のパーミッションは **rw-** である。
- **susan** ユーザーのパーミッションは **---** である。
- 他のユーザーのパーミッションは **r--** である。

手順

```
# setfacl -m u:andrew:rw- group-project
# setfacl -m u:susan:--- group-project
```

検証手順

- ユーザー **andrew** に **rw-** パーミッションがあり、ユーザー **susan** には **---** パーミッションがあり、その他のユーザーに **r--** パーミッションがあることを確認するには、以下を実行します。

```
$ getfacl group-project
```

返される出力は以下のとおりです。

```
# file: group-project
# owner: root
# group: root
```

```
user:andrew:rw-
user:susan:---
group::r--
mask::rw-
other::r--
```

11.3. UMASK の管理

umask ユーティリティーを使用して、umask の現在の値またはデフォルト値を表示、設定、または変更できます。

11.3.1. umask の現在の値の表示

umask ユーティリティーを使用して、umask の現在の値をシンボリックモードまたは 8 進数モードで表示できます。

手順

- umask の現在の値をシンボリックモードで表示するには、以下のコマンドを使用します。

```
$ umask -S
```

- umask の現在の値を 8 進法で表示するには、以下のコマンドを使用します。

```
$ umask
```

注記

umask を 8 進法で表示するには、4 桁の数字 (**0002** または **0022**) で表示される場合があります。**umask** の最初の数字は、特殊ビット (スティッキービット、SGID ビット、または SUID ビット) を表します。最初の数字を **0** に設定すると、特別なビットは設定されません。

11.3.2. デフォルトの bash umask の表示

bash、**ksh**、**zsh**、**tcsh** などの多くのシェルを使用できます。これらのシェルはログインまたは nologin シェルとして動作します。ネイティブまたは GUI 端末を開いて、ログインシェルを呼び出すことができます。

ログインシェルまたは nologin シェルのどちらでコマンドを実行しているかを確認するには、**echo $0** コマンドを使用します。

例11.8 ログインまたは nologin bash シェルで作業しているかどうかの確認

- **echo $0** コマンドの出力が **bash** を返す場合、nologin シェルでコマンドを実行します。

```
$ echo $0
bash
```

nologin シェルのデフォルトの **umask** は、**/etc/bashrc** 設定ファイルで設定します。

- **echo $0** コマンドの出力が **-bash** を返す場合は、ログインシェルでコマンドを実行します。

```
# echo $0
-bash
```

ログインシェルのデフォルトの **umask** は **/etc/login.defs** 設定ファイルで設定します。

手順

- nologin シェルのデフォルトの **bash umask** を表示するには、以下のコマンドを使用します。

```
$ grep umask /etc/bashrc
```

返される出力は以下のとおりです。

```
# By default, we want umask to get set. This sets it for non-login shell.
      umask 002
      umask 022
```

- ログインシェルのデフォルトの **bash umask** を表示するには、以下のコマンドを使用します。

```
$ grep "UMASK" /etc/login.defs
```

返される出力は以下のとおりです。

```
# 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.
```

11.3.3. シンボリック値を使用した umask の設定

シンボリック値 (組み合わせ文字および記号) を指定して **umask** ユーティリティーを使用し、現在のシェルセッションの **umask** を設定できます。

以下の **パーミッション** を割り当てることができます。

- 読み取り (**r**)
- 書き込み (**w**)
- 実行 (**x**)

パーミッションは、以下の **レベルの所有権** に割り当てることができます。

- ユーザー所有者 (**u**)
- グループ所有者 (**g**)
- その他 (**o**)
- すべて (**a**)

パーミッションを追加または削除するには、以下の **記号** を使用できます。

- **+**: 既存のパーミッションの上にパーミッションを追加します。
- **-**: 既存のパーミッションからパーミッションを削除します。
- **=**: 既存のパーミッションを削除し、新しいパーミッションを明示的に定義します。

注記

パーミッションを等号 (**=**) の後ろに指定していない場合には自動的に無視されます。

手順

- 現在のシェルセッションの **umask** を設定するには、以下のコマンドを使用します。

```
$ umask -S <level><operation><permission>
```

<level> は、umask を設定する 所有権のレベル に置き換えます。**<operation>** は、署名 の 1 つに置き換えます。**<permission>** は、割り当てる パーミッション に置き換えます。たとえば、umask を **u=rwx,g=rwx,o=rwx** に設定するには **umask -S a=rwx** を使用します。

詳細は、ユーザーファイル作成モードを参照してください。

注記

umask は、現在のシェルセッション限定で有効になります。

11.3.4. 8 進数値を使用した umask の設定

8 進数値 (数字) を指定して **umask** ユーティリティーを使用し、現在のシェルセッションの **umask** を設定できます。

手順

- 現在のシェルセッションの **umask** を設定するには、以下のコマンドを使用します。

```
$ umask octal_value
```

octal_value は 8 進数値に置き換えます。詳細は、 ユーザーファイル作成モードマスクを参照してください。

注記

umask は、現在のシェルセッション限定で有効になります。

11.3.5. nologin シェルのデフォルト umask の変更

/etc/bashrc ファイルを変更して、標準ユーザーのデフォルトの **bash** umask を変更できます。

前提条件

- **root** アクセス

手順

1. **root** として、エディターで **/etc/bashrc** ファイルを開きます。
2. 以下のセクションを変更して、新しいデフォルトの **bash umask** を設定します。

```
if [ $UID -gt 199 ] && [ “id -gn” = “id -un” ]; then
   umask 002
else
   umask 022
fi
```

   umask (**002**) のデフォルト値を別の進数値に置き換えます。詳細は、ユーザーファイル作成モードマスクを参照してください。

3. 変更を保存し、エディターを終了します。

11.3.6. ログインシェルのデフォルト umask の変更

/etc/login.defs ファイルを変更して、**root** ユーザーのデフォルトの **bash umask** を変更できます。

前提条件

- **root** アクセス

手順

1. **root** として、エディターで **/etc/login.defs** ファイルを開きます。
2. 以下のセクションを変更して、新しいデフォルトの **bash umask** を設定します。

```
# 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
```

   umask (**022**) のデフォルト値を別の 8 進数値に置き換えます。詳細は、ユーザーファイル作成モードマスクを参照してください。

3. 変更を保存し、エディターを終了します。

11.3.7. 特定ユーザーのデフォルトの umask の変更

特定ユーザーのデフォルトの **umask** を変更するには、そのユーザーの **.bashrc** を変更します。

手順

- umask の 8 進数値を指定する行を、特定ユーザーの **.bashrc** ファイルに追加します。

```
$ echo 'umask octal_value' >> /home/username/.bashrc
```

octal_value は 8 進数値に、 username はユーザー名に置き換えます。詳細は、ユーザーファイル作成モードマスクを参照してください。

11.3.8. 新しく作成されたホームディレクトリーのデフォルト権限設定

新しく作成されたユーザーのホームディレクトリーのパーミッションモードは、**/etc/login.defs** ファイルを修正して変更できます。

手順

1. **root** として、エディターで **/etc/login.defs** ファイルを開きます。
2. 以下のセクションを変更して、**HOME_MODE** のデフォルトを新規設定します。

```
# 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
```

デフォルトの 8 進数値 (**0700**) を別の 8 進数値に置き換えます。選択したモードは、ホームディレクトリーのパーミッションの作成に使用されます。

3. **HOME_MODE** が設定されている場合は、変更を保存してエディターを終了します。
4. **HOME_MODE** が設定されていない場合は、**UMASK** を変更して、新しく作成されたホームディレクトリーにモードを設定します。

```
# 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
```

デフォルトの 8 進数値 (**022**) を別の 8 進数値に置き換えます。詳細は、ユーザーファイル作成モードマスクを参照してください。

5. 変更を保存し、エディターを終了します。

第12章 SYSTEMD の管理

システム管理者は、**systemd** を使用してシステムの重要な側面を管理できます。Linux オペレーティングシステムのシステムおよびサービスマネージャーとして機能する **systemd** ソフトウェアスイートは、制御、レポート、およびシステム初期化のためのツールとサービスを提供します。**systemd** の主な機能は次のとおりです。

- ブート時のシステムサービスの並行起動
- デーモンのオンデマンドアクティベーション
- 依存関係ベースのサービス制御ロジック

systemd が管理する基本オブジェクトは、システムのリソースとサービスを表す **systemd ユニット** です。**systemd** ユニットは、名前、タイプ、および特定のタスクを定義および管理する設定ファイルで構成されます。ユニットファイルを使用すると、システムの動作を設定できます。さまざまな systemd ユニットタイプの例を以下に示します。

サービス
個々のシステムサービスを制御および管理します。

ターゲット
システム状態を定義するユニットのグループを表します。

デバイス
ハードウェアデバイスとその可用性を管理します。

マウント
ファイルシステムのマウントを処理します。

Timer
タスクを特定の間隔で実行するようにスケジュールします。

注記

利用可能なすべてのユニットタイプを表示するには、以下を実行します。

```
# systemctl -t help
```

12.1. SYSTEMD のユニットファイルの場所

ユニット設定ファイルは、次のいずれかのディレクトリーにあります。

表12.1 systemd のユニットファイルの場所

ディレクトリー	説明
/usr/lib/systemd/system/	インストール済みの RPM パッケージで配布された **systemd** のユニットファイル。
/run/systemd/system/	ランタイム時に作成された **systemd** ユニットファイル。このディレクトリーは、インストール済みのサービスのユニットファイルのディレクトリーよりも優先されます。

ディレクトリー	説明
/etc/systemd/system/	**systemctl enable** コマンドを使用して作成された **systemd** ユニットファイルと、サービスを拡張するために追加されたユニットファイル。このディレクトリーは、runtime のユニットファイルのディレクトリーよりも優先されます。

systemd のデフォルト設定はコンパイル中に定義され、**/etc/systemd/system.conf** ファイルで確認できます。このファイルを編集すると、**systemd** ユニットの値をシステム全体でオーバーライドしてデフォルト設定を変更できます。

たとえば、タイムアウト制限のデフォルト値 (90 秒) を上書きする場合は、**DefaultTimeoutStartSec** パラメーターを使用して、上書きする値を秒単位で入力します。

```
DefaultTimeoutStartSec=required value
```

12.2. SYSTEMCTL によるシステムサービス管理

システム管理者は、**systemctl** ユーティリティーを使用してシステムサービスを管理できます。実行中のサービスの起動、停止、再起動、ブート時に起動するサービスの有効化と無効化、利用可能なサービスのリスト表示、システムサービスのステータスの表示など、さまざまなタスクを実行できます。

12.2.1. システムサービスのリスト表示

現在ロードされているすべてのサービスユニットをリストし、使用可能なすべてのサービスユニットのステータスを表示できます。

手順

systemctl コマンドを使用して、次のタスクのいずれかを実行します。

- 現在ロードされているすべてのサービスユニットをリストします。

```
$ systemctl list-units --type service
UNIT                    LOAD   ACTIVE SUB     DESCRIPTION
abrt-ccpp.service       loaded active exited  Install ABRT coredump hook
abrt-oops.service       loaded active running ABRT kernel log watcher
abrtd.service           loaded active running ABRT Automated Bug Reporting Tool
...
systemd-vconsole-setup.service loaded active exited  Setup Virtual Console
tog-pegasus.service           loaded active running OpenPegasus CIM Server

LOAD   = Reflects whether the unit definition was properly loaded.
ACTIVE = The high-level unit activation state, or a generalization of SUB.
SUB    = The low-level unit activation state, values depend on unit type.

46 loaded units listed. Pass --all to see loaded but inactive units, too.
To show all installed unit files use 'systemctl list-unit-files'
```

デフォルトでは、**systemctl list-units** コマンドは、アクティブなユニットのみを表示します。このコマンドは、サービスユニットファイルごとに、次のパラメーターの概要を提供します。

UNIT

サービスユニットのフルネーム

LOAD

設定ファイルのロード状態

ACTIVE または SUB

現在の高レベルおよび低レベルのユニットファイルのアクティベーション状態

DESCRIPTION

ユニットの目的と機能の簡単な説明

- **--all** または **-a** コマンドラインオプションを指定して次のコマンドを使用し、**ロードされたすべてのユニットを状態に関係なく** をリスト表示します。

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

- 利用可能なすべてのサービスユニットのステータス (**enabled** または **disabled**) をリスト表示します。

```
$ systemctl list-unit-files --type service
UNIT FILE                              STATE
abrt-ccpp.service                        enabled
abrt-oops.service                        enabled
abrtd.service                          enabled
...
wpa_supplicant.service                       disabled
ypbind.service                         disabled

208 unit files listed.
```

このコマンドでは、サービスユニットごとに以下を表示します。

UNIT FILE

サービスユニットのフルネーム

STATE

サービスユニットがブート時に自動的に起動するかどうかの情報

関連情報

- システムサービスステータスの表示

12.2.2. システムサービスステータスの表示

サービスユニットを検査して詳細情報を取得し、サービスの状態 (ブート時の起動が有効かどうか、現在実行中かどうか) を確認できます。特定のサービスユニットの前または後に起動するように指定されたサービスを表示することもできます。

手順

systemctl コマンドを使用して、次のタスクのいずれかを実行します。

- システムサービスに対応するサービスユニットに関する詳細情報を表示します。

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

<name> は、確認するサービスユニットの名前 (**gdm** など) に置き換えます。

このコマンドでは、以下の情報が表示されます。

- 選択したサービスユニットの名前とその後に続く簡単な説明
- 利用可能なサービスユニットの情報 で説明されている 1 つ以上のフィールド
- サービスユニットの実行: ユニットが **root** ユーザーによって実行される場合
- 最新のログエントリー

表12.2 利用可能なサービスユニットの情報

フィールド	説明
Loaded	サービスユニットがロードされているかどうかの説明、ユニットファイルへの絶対パス、およびブート時のユニット起動が有効かどうかの注記。
Active	サービスユニットが実行中かどうかの説明と、タイムスタンプ
Main PID	プロセス ID と、対応するシステムサービスの名前。
ステータス	対応するシステムサービスに関する追加情報
Process	関連プロセスに関する追加情報
CGroup	関連するコントロールグループ (**cgroups**) に関する追加情報。

例12.1 サービスステータスの表示

GNOME Display Manager のサービスユニット名は **gdm.service** になります。このサービスユニットの現在のステータスを確認するには、シェルプロンプトで次のコマンドを実行します。

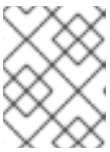
```
# systemctl status gdm.service
gdm.service - GNOME Display Manager
   Loaded: loaded (/usr/lib/systemd/system/gdm.service; enabled)
   Active: active (running) since Thu 2013-10-17 17:31:23 CEST; 5min ago
 Main PID: 1029 (gdm)
   CGroup: /system.slice/gdm.service
           ├─1029 /usr/sbin/gdm
           └─1047 /usr/bin/Xorg :0 -background none -verbose -auth /r...

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

- 特定のサービスユニットが実行中であることを確認します。

```
$ systemctl is-active <name>.service
```

- 特定のサービスユニットのブート時起動が有効かどうかを確認します。

```
$ systemctl is-enabled <name>.service
```



注記

systemctl is-active および **systemctl is-enabled** コマンドは、指定したサービスユニットが実行中または有効な場合に、終了ステータス **0** を返します。

- 指定したサービスユニットの前に **systemd** がどのサービスの起動を指示するかを確認します。

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

たとえば、**gdm** の前に起動するサービスのリストを表示するには、次のように入力します。

```
# systemctl list-dependencies --after gdm.service
gdm.service
├─dbus.socket
├─getty@tty1.service
├─livesys.service
├─plymouth-quit.service
├─system.slice
├─systemd-journald.socket
├─systemd-user-sessions.service
└─basic.target
[output truncated]
```

- 指定したサービスユニットの後に **systemd** がどのサービスの起動を指示するかを確認します。

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

たとえば、**gdm** の後に起動するように **systemd** が指示するサービスのリストを表示するには、次のように入力します。

```
# systemctl list-dependencies --before gdm.service
gdm.service
├─dracut-shutdown.service
├─graphical.target
│ ├─systemd-readahead-done.service
│ ├─systemd-readahead-done.timer
│ └─systemd-update-utmp-runlevel.service
└─shutdown.target
  ├─systemd-reboot.service
  └─final.target
    └─systemd-reboot.service
```

関連情報

- システムサービスのリスト表示

12.2.3. システムサービスの起動

start コマンドを使用すると、現在のセッションでシステムサービスを起動できます。

前提条件

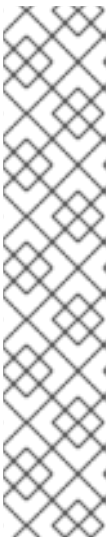
- Root アクセス

手順

- 現在のセッションでシステムサービスを起動します。

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

<name> は、起動するサービスユニットの名前 (**httpd.service** など) に置き換えます。



注記

systemd には、サービス間で正と負の依存関係が存在します。特定のサービスを起動するとき、別のサービスを 1 つまたは複数開始 (**正の依存関係**)、あるいはサービスを 1 つまたは複数停止 (**負の依存関係**) することが必要となる場合があります。

新しいサービスの起動を試みると、ユーザーに明示的な通知なしに、**systemd** がすべての依存関係を自動的に解決します。つまり、サービスを実行していて、負の依存関係にある別のサービスを起動しようとすると、最初のサービスが自動的に停止します。

たとえば、**postfix** サービスを実行している時に **sendmail** サービスを起動すると、**systemd** は、自動的に **postfix** を停止します。この 2 つのサービスは競合するため、同じポートでは実行できません。

関連情報

- **systemctl(1)** man ページ
- ブート時のシステムサービス起動の有効化
- システムサービスステータスの表示

12.2.4. システムサービスの停止

現行セッションでシステムサービスを停止するには、**stop** コマンドを使用します。

前提条件

- Root アクセス

手順

- システムサービスを停止します。

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

<name> は、停止するサービスユニットの名前 (**bluetooth** など) に置き換えます。

関連情報

- **systemctl(1)** man ページ
- ブート時のシステムサービス起動の無効化
- システムサービスステータスの表示

12.2.5. システムサービスの再起動

restart コマンドを使用して次のアクションを実行すると、現在のセッションでシステムサービスを再起動できます。

- 現在のセッションで選択したサービスユニットを停止し、すぐに再起動する。
- 対応するサービスがすでに実行中の場合にのみ、サービスユニットを再起動する。
- システムサービスの実行を中断せずに、システムサービスの設定を再ロードする。

前提条件

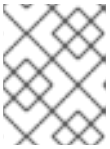
- Root アクセス

手順

- システムサービスを再起動します。

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

<name> は、再起動するサービスユニットの名前 (**httpd** など) に置き換えます。



注記

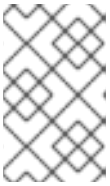
選択したサービスユニットが実行中でない場合には、このコマンドでこのサービスユニットが起動します。

- オプション: 対応するサービスがすでに実行中の場合にのみ、サービスユニットを再起動します。

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

- オプション: サービスの実行を中断せずに設定を再ロードします。

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



注記

システムサービスがこの機能をサポートしない場合は、このコマンドは無視されることに注意してください。このようなサービスを再起動するには、代わりに **reload-or-restart** コマンドおよび **reload-or-try-restart** コマンドを使用します。

関連情報

- **systemctl** の man ページ
- システムサービスステータスの表示

12.2.6. ブート時のシステムサービス起動の有効化

ブート時のサービスの自動起動を有効にすることができます。この変更は次回のリブート時に適用されます。

前提条件

- Root アクセス
- 有効にするサービスがマスクされていない。マスクされたサービスがある場合は、まずマスクを解除します。

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

手順

- システムの起動時にサービスが起動するようにします。

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

<name> は、有効にするサービスユニット名 (**httpd** など) に置き換えます。

- オプション: 1 つのコマンドでサービスを有効にして起動することもできます。

```
# systemctl enable --now <name>.service
```

関連情報

- **systemctl (1)** man ページ
- システムサービスステータスの表示
- システムサービスの起動

12.2.7. ブート時のシステムサービス起動の無効化

システムの起動時にサービスユニットが自動的に起動しないようにすることができます。サービスを無効にすると、ブート時に起動されませんが、手動で起動できます。手動で開始できないようにサービスをマスクすることもできます。マスキングは、サービスが再度マスク解除されるまでサービスが永続的に使用できなくなるようにするサービスを無効にする方法です。

前提条件

- Root アクセス

手順

- サービスがブート時に起動するのを無効にします。

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

<name> は、無効にするサービスユニットの名前 (**bluetooth** など) に置き換えます。

- オプション: サービスを永久に使用不可にする場合は、サービスをマスクします。

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

このコマンドにより、**/etc/systemd/system/name.service** ファイルを、**/dev/null** へのシンボリックリンクに置き換え、実際のユニットファイルが **systemd** ファイルにアクセスできないようにします。

関連情報

- **systemctl (1)** man ページ
- システムサービスステータスの表示
- システムサービスの停止

12.3. ターゲットシステム状態でのブート

システム管理者は、システムのブートプロセスを制御し、システムがブートする状態を定義できます。これは **systemd** ターゲットと呼ばれ、特定のレベルの機能を達成するためにシステムが起動する **systemd** ユニットのセットです。systemd ターゲットの操作時には、デフォルトのターゲットの表示、実行時のターゲットの選択、デフォルトのブートターゲットの変更、緊急ターゲットまたはレスキューターゲットでのブートを行うことができます。

12.3.1. ターゲットユニットファイル

systemd ターゲットは、システムの起動時に同期ポイントとして機能する関連ユニットのグループです。**.target** ファイル拡張子で終わるターゲットユニットファイルは、**systemd** ターゲットを表します。ターゲットユニットの目的は、依存関係のチェーンでさまざまな **systemd** ユニットをグループ化することです。

以下の例を参照してください。

- グラフィカルセッションを開始するための **graphical.target unit** は、GNOME Display Manager (**gdm.service**)または Accounts Service (**accounts-daemon.service**)などのシステムサービスを開始し、**multi-user.target unit** もアクティブにします。
- 同様に、**multi-user.target** ユニットは、NetworkManager (**NetworkManager.service**)、D-Bus (**dbus.service**) といった、その他の必須システムサービスを開始し、**basic.target** という別のターゲットユニットをアクティブにします。

次の **systemd** ターゲットをデフォルトまたは現在のターゲットとして設定できます。

表12.3 一般的な systemd ターゲット

rescue	ベースシステムにプルしてレスキューシェルを生成するユニットターゲット
multi-user	マルチユーザーシステムを設定するためのユニットターゲット
graphical	グラフィカルログイン画面を設定するためのユニットターゲット
emergency	メインコンソールで緊急シェルを起動するユニットターゲット

関連情報

- **systemd.special(7)** man ページ
- **systemd.target(5)** man ページ

12.3.2. ブート先のデフォルトターゲットの変更

システムが起動すると、**systemd** は、実際のターゲットユニットを指す **default.target** シンボリックリンクをアクティブにします。現在選択されているデフォルトのターゲットユニットは、**/etc/systemd/system/default.target** ファイルで確認できます。各ターゲットは特定のレベルの機能を表し、他のユニットをグループ化するために使用されます。さらに、ターゲットユニットはブート時に同期ポイントとして機能します。システムがブートするデフォルトターゲットを変更できます。デフォルトのターゲットユニットを設定すると、次回の再起動まで現在のターゲットは変更されません。

前提条件

- Root アクセス

手順

1. **systemd** がシステムを起動するために使用する現在のデフォルトのターゲットユニットを確認します。

```
# systemctl get-default
graphical.target
```

2. 現在ロードされているターゲットをリストします。

```
# systemctl list-units --type target
```

3. 別のターゲットユニットをデフォルトで使用するようにシステムを設定します。

```
# systemctl set-default <name>.target
```

<name> は、デフォルトで使用するターゲットユニットの名前に置き換えます。

```
Example:
# systemctl set-default multi-user.target
Removed /etc/systemd/system/default.target
Created symlink /etc/systemd/system/default.target -> /usr/lib/systemd/system/multi-user.target
```

–

4. デフォルトのターゲットユニットを確認します。

```
# systemctl get-default
multi-user.target
```

5. リブートして変更を適用します。

```
# reboot
```

関連情報

- **systemctl(1)** man ページ
- **systemd.special(7)** man ページ
- **bootup(7)** man ページ

12.3.3. 現在のターゲットの変更

実行中のシステムでは、リブートせずに現在のブートでターゲットユニットを変更できます。別のターゲットに切り替えると、**systemd** は、このターゲットが必要とするすべてのサービスとその依存関係を起動し、新しいターゲットで有効になっていないすべてのサービスを停止します。別のターゲットの分離は、現在のブートにのみ影響します。

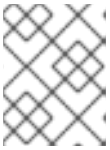
手順

1. オプション: 現在のターゲットを確認します。

```
# systemctl get-default
graphical.target
```

2. オプション: 選択できるターゲットのリストを表示します。

```
# systemctl list-units --type target
```



注記

ユニットファイルに **AllowIsolate=yes** オプションが設定されているターゲットのみを分離できます。

3. 現在のブートで別のターゲットユニットに変更します。

```
# systemctl isolate <name>.target
```

<name> は、現在のブートで使用するターゲットユニットの名前に置き換えます。

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

このコマンドは、**multi-user** という名前のターゲットユニットとすべての従属ユニットを起動し、他のすべてのユニットをただちに停止します。

関連情報

- **systemctl(1)** man ページ

12.3.4. レスキューモードでの起動

システムが後のターゲットにアクセスできず、通常のブートプロセスが失敗した場合に、トラブルシューティングまたは修復のためのシングルユーザー環境を提供する **レスキューモード** で起動できます。レスキューモードでは、システムはすべてのローカルファイルシステムをマウントし、特定の重要なシステムサービスを起動しようとしますが、ネットワークインターフェイスはアクティブになりません。

前提条件

- Root アクセス

手順

- レスキューモードに入るには、現行セッションで現在のターゲットを変更します。

```
# systemctl rescue

Broadcast message from root@localhost on pts/0 (Fri 2023-03-24 18:23:15 CEST):

The system is going down to rescue mode NOW!
```

注記

このコマンドは **systemctl isolate rescue.target** と似ていますが、システムに現在ログインしているすべてのユーザーに情報メッセージを送信します。

systemd がメッセージを送信しないようにするには、**--no-wall** コマンドラインオプションを指定して次のコマンドを入力します。

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

トラブルシューティングの手順

システムがレスキューモードに移行できない場合は、可能な限り最小限の環境を提供する **緊急モード** でブートできます。緊急モードでは、システムは root ファイルシステムを読み込み専用でマウントし、他のローカルファイルシステムのマウントは試みません。また、ネットワークインターフェイスのアクティブ化も行わず、限定的な必須サービスのみを起動します。

12.3.5. ブートプロセスのトラブルシューティング

システム管理者は、ブート時にデフォルト以外のターゲットを選択して、ブートプロセスのトラブルシューティングを行うことができます。ブート時のターゲットの変更は、1回のブートにしか影響しません。可能な限り最小限の環境を提供する **緊急モード** で起動できます。

手順

1. システムを再起動し、通常のブートを開始する Enter キー以外の任意のキーを押してブートローダーメニューのカウントダウンを中断します。

2. 開始するカーネルエントリーにカーソルを移動します。
3. E キーを押して、現在のエントリーを編集します。
4. **linux** で始まる行の末尾に移動し、Ctrl+E を押して行の末尾にジャンプします。

```
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
```

5. 別のブーターゲットを選択するには、**linux** で始まる行の末尾に **systemd.unit=** パラメーターを追加します。

```
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
systemd.unit=<name>.target
```

   <name> は、使用するターゲットユニットの名前に置き換えます。たとえば、**systemd.unit=emergency.target** です。

6. Ctrl+X を押して、これらの設定で起動します。

12.4. システムのシャットダウン、サスペンド、およびハイバネート

システム管理者は、さまざまな電源管理オプションを使用して電力消費を管理したり、適切なシャットダウンを実行してすべてのデータを確実に保存したり、システムを再起動して変更や更新を適用したりできます。

12.4.1. システムのシャットダウン

システムをシャットダウンする場合は、**systemctl** ユーティリティーを使用するか、**shutdown** コマンドを使用してこのユーティリティーを呼び出します。

shutdown を使用すると、次の利点があります。

- **time** 引数を使用してシャットダウンをスケジュールできます。また、この引数を使用すると、システムのシャットダウンがスケジュールされていることがユーザーに警告されます。
- シャットダウンをキャンセルできます。

関連情報

- systemctl を使用した電源管理コマンドの概要

12.4.2. システムのシャットダウンのスケジュール設定

システム管理者は、ユーザーが作業内容を保存してシステムからログオフする時間を与えるために、遅延シャットダウンをスケジュールできます。**shutdown** コマンドを使用して、次の操作を実行します。

- 特定の時刻にシステムをシャットダウンし、マシンの電源をオフにする
- マシンの電源を切らずにシステムをシャットダウンして停止する
- 保留中のシャットダウンをキャンセルする

前提条件

- Root アクセス

手順

shutdown コマンドを使用して、次のタスクのいずれかを実行します。

- システムをシャットダウンしてマシンの電源をオフにする時刻を指定します。

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

hh:mm は 24 時間表記の時刻です。新しいログインを防ぐために、システムがシャットダウンする 5 分前に **/run/nologin** ファイルが作成されます。

time 引数を使用すると、オプションの **ウォールメッセージ** を指定して、システムにログインしているユーザーにシャットダウンの予定を通知できます。たとえば、**shutdown --poweroff 13:59 "Attention.The system will shut down at 13:59"** などのメッセージです。

- マシンの電源を切らずに、時間をおいてからシステムをシャットダウンして停止します。

```
# shutdown --halt +m
```

+m は遅らせる時間 (分) です。 **now** キーワードを **+0** のエイリアスとして使用できます。

- 保留中のシャットダウンをキャンセルします。

```
# shutdown -c
```

関連情報

- **shutdown(8)** の man ページ
- systemctl コマンドを使用したシステムのシャットダウン

12.4.3. systemctl コマンドを使用したシステムのシャットダウン

システム管理者は、**systemctl** コマンドを使用して、システムをシャットダウンしてマシンの電源をオフにしたり、マシンの電源をオフにせずにシステムをシャットダウンして停止したりできます。

前提条件

- Root アクセス

手順

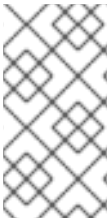
systemctl コマンドを使用して、次のタスクのいずれかを実行します。

- システムをシャットダウンし、マシンの電源をオフにします。

```
# systemctl poweroff
```

- マシンの電源を切らずにシステムをシャットダウンして停止します。

```
# systemctl halt
```



注記

デフォルトでは、これらのコマンドのいずれかを実行すると、**systemd** が現在システムにログインしているすべてのユーザーに情報メッセージを送信します。**systemd** がメッセージを送信しないようにするには、コマンドラインオプション **--no-wall** を付けてコマンドを実行します。

12.4.4. システムの再起動

システムを再起動すると、**systemd** は実行中のすべてのプログラムとサービスを停止します。システムはシャットダウンすると、すぐに再起動します。システムの再起動は、次の場合に役立ちます。

- 新しいソフトウェアまたは更新をインストールした後
- システム設定を変更した後
- システムの問題のトラブルシューティングを行う場合

前提条件

- Root アクセス

手順

- システムを再起動します。

```
# systemctl reboot
```

注記

デフォルトでは、このコマンドを使用すると、**systemd** が現在システムにログインしているすべてのユーザーに情報メッセージを送信します。**systemd** がこのメッセージを送信しないようにするには、**--no-wall** オプションを指定してこのコマンドを実行します。

12.4.5. システムのサスペンドおよびハイバネートによる電力消費の最適化

システム管理者は、電力消費を管理し、システムのエネルギーを節約し、システムの現在の状態を保存できます。これを行うには、次のモードのいずれかを適用します。

サスペンド

サスペンドは、システムの状態を RAM に保存し、マシンにある、RAM モジュール以外のほとんどのデバイスの電源を切ります。マシンの電源を戻すと、システムは再起動せずに RAM からその状態を復元します。システムの状態がハードディスクではなくメモリーに保存されるため、システムは、ハイバネートよりも、サスペンドモードからのほうがはるかに早く復元できます。ただし、サスペンドされたシステム状態は停電に対して脆弱です。

ハイバネート

ハイバネートは、システムの状態をハードディスクドライブに保存し、マシンの電源を切ります。マシンの電源を戻すと、システムは再起動せずに、保存されたデータからその状態を復元します。システムの状態がメモリーではなくハードディスクに保存されるため、マシンでメモリーモジュールへの電源供給を維持する必要はありません。ただし、システムは、サスペンドモードより、ハイバネーションから復元する方がはるかに遅くなります。

ハイブリッドスリープ

これは、ハイバネートとサスペンドの両方の要素を組み合わせたものです。システムはまず現在の

状態をハードディスクドライブに保存し、サスペンドと同様の低電力状態に入ります。これにより、システムはより迅速に再開できるようになります。ハイブリッドスリープの利点は、スリープ状態中にシステムの電源が失われた場合でも、ハイバネーションと同様に、ハードディスクに保存されたイメージから以前の状態を回復できることです。

サスペンドしてからハイバネート

このモードでは、まずシステムがサスペンドされます。これにより、現在のシステムの状態が RAM に保存され、システムが低電力モードになります。一定期間 (**HibernateDelaySec** パラメーターで定義可能) サスペンド状態が続くと、システムはハイバネートします。ハイバネーションは、システムの状態をハードディスクドライブに保存し、システムを完全にシャットダウンします。サスペンドしてからハイバネートするモードには、バッテリー電力を節約しながら、作業をすぐに再開できるという利点があります。さらに、このモードでは、停電に備えてデータが確実に保存されます。

前提条件

- Root アクセス

手順

適切な省電力方法を選択します。

- システムをサスペンドします。

```
# systemctl suspend
```

- システムをハイバネートします。

```
# systemctl hibernate
```

- システムをハイバネートおよびサスペンドします。

```
# systemctl hybrid-sleep
```

- システムをサスペンドしてからハイバネートします。

```
# systemctl suspend-then-hibernate
```

12.4.6. systemctl を使用した電源管理コマンドの概要

以下の **systemctl** コマンドを使用して、システムの電源管理を制御できます。

表12.4 systemctl 電源管理コマンドの概要

systemctl コマンド	説明
systemctl halt	システムを停止します。
systemctl poweroff	システムの電源を切ります。
systemctl reboot	システムを再起動します。
systemctl suspend	システムをサスペンドします。

systemctl コマンド	説明
systemctl hibernate	システムをハイバネートします。
systemctl hybrid-sleep	システムをハイバネートおよびサスペンドします。

12.4.7. 電源ボタンの動作の変更

コンピューターの電源ボタンを押すと、デフォルトではシステムが一時停止またはシャットダウンします。この動作は好みに応じてカスタマイズできます。

12.4.7.1. systemd の電源ボタンの動作の変更

非グラフィカルな **systemd** ターゲットで電源ボタンを押すと、デフォルトではシステムがシャットダウンします。この動作は好みに応じてカスタマイズできます。

前提条件

- 管理アクセスがある。

手順

1. **/etc/systemd/logind.conf** 設定ファイルを開きます。
2. **HandlePowerKey=poweroff** という行を探します。
3. 行が **#** 記号で始まる場合は、この記号を削除して設定を有効にします。
4. **poweroff** を次のオプションのいずれかに置き換えます。

   poweroff
   コンピューターをシャットダウンします。
   reboot
   システムを再起動します。
   halt
   システムの停止を開始します。
   kexec
   kexec の再起動を開始します。
   suspend
   システムをサスペンドします。
   hibernate
   システムのハイバネートを開始します。
   ignore
   なにもしない

   たとえば、電源ボタンを押したときにシステムを再起動するには、次の設定を使用します。

   ```
   HandlePowerKey=reboot
   ```

5. 変更を保存してエディターを閉じます。

次のステップ

- グラフィカルセッションを使用する場合は、GNOME で電源ボタンも設定します。「GNOME での電源ボタンの動作の変更」 を参照してください。

12.4.7.2. GNOME での電源ボタンの動作の変更

グラフィカルログイン画面またはグラフィカルユーザーセッションで、電源ボタンを押すと、デフォルトでマシンがサスペンドされます。これはユーザーが物理的に電源ボタンを押した場合と、リモートコンソールから仮想の電源ボタンを押した場合の両方で起きます。電源ボタンの動作は、別のものを選択することもできます。

前提条件

- **systemd** で電源ボタンの動作を設定しました。「systemd の電源ボタンの動作の変更」 を参照してください。

手順

1. システム全体の設定用に、ローカルデータベースを **/etc/dconf/db/local.d/01-power** に作成します。以下のコンテンツを入力します。

```
[org/gnome/settings-daemon/plugins/power]
power-button-action='suspend'
```

suspend を次の電源ボタンのアクションのいずれかに置き換えます。

nothing
何も実行しません。

suspend
システムをサスペンドします。

hibernate
システムをハイバネートします。

interactive
何を実行するかをユーザーに質問するポップアップクエリーを表示します。
インタラクティブモードでは、電源ボタンを押すと、60 秒後に自動的にシステムの電源がオフになります。ただし、ポップアップクエリーとは異なる動作を選択できます。

2. オプション: ユーザーの設定を上書きし、ユーザーが設定を変更できないようにします。**/etc/dconf/db/local.d/locks/01-power ファイルに以下の設定を入力します。**

```
/org/gnome/settings-daemon/plugins/power/power-button-action
```

3. システムデータベースを更新します。

```
# dconf update
```

4. システム全体の設定を有効にするには、ログアウトしてから再度ログインします。

第13章 時刻同期の設定

IT 環境では、正確な時間管理が重要です。すべてのネットワークデバイスで時刻が一貫していれば、ログファイルのトレーサビリティーが向上します。また、特定のプロトコルは同期されたクロックを使用します。たとえば、Kerberos はタイムスタンプを使用してリプレイ攻撃を防ぎます。

13.1. CHRONY スイートを使用した NTP の設定

IT では、複数の理由から、正確な時間管理が重要です。たとえばネットワーキングでは、パケットとログのタイムスタンプが正確であることが必要になります。Linux システムでは、**NTP** プロトコルがユーザー領域で実行しているデーモンにより実装されます。

ユーザー領域のデーモンは、カーネルで実行しているシステムクロックを更新します。システムクロックは、さまざまなクロックソースを使用して時間を維持します。一般的に使用されるのは **Time Stamp Counter** (**TSC**) です。TSC は、最後にリセットされた時点からのサイクル数を計測する CPU レジスターです。非常に高速でハイレゾリューションであり、中断も発生しません。

Red Hat Enterprise Linux 8 以降、**NTP** プロトコルは **chronyd** デーモンにより実装されます。このデーモンは、**chrony** パッケージのリポジトリーから利用できます。

次のセクションでは、**chrony** スイートを使用して NTP を設定する方法を説明します。

13.1.1. chrony スイートの概要

chrony は **Network Time Protocol (NTP)** の実装です。**chrony** を使用すると、以下のことができます。

- システムクロックを、**NTP** サーバーと同期する
- システムクロックを、GPS レシーバーなどの基準クロックと同期する
- システムクロックを、手動で入力した時間と同期する
- ネットワーク内の他のコンピューターにタイムサービスを提供する **NTPv4(RFC 5905)** サーバーまたはピアとして

chrony は、ネットワーク接続が頻繁に切断される、ネットワークの混雑が長時間続く、温度が変わる (一般的なコンピューターのクロックは温度に敏感) といったさまざまな状況や、継続して実行されない、または仮想マシンで実行されているといったシステムにおいても、良好に動作します。

インターネット上で同期している 2 つのマシン間の一般的精度は数ミリ秒以内、LAN 上のマシン間では数十マイクロ秒以内です。ハードウェアのタイムスタンプまたはハードウェア基準クロックは、同期している 2 つのマシン間の精度をサブマイクロ秒レベルにまで高めることができます。

chrony は、ユーザー空間で実行する **chronyd** と、**chronyd** のパフォーマンスを監視し、実行時にさまざまなオペレーティングパラメーターを変更するのに使用できるコマンドラインプログラムである **chronyc** で構成されます。

chrony デーモンである **chronyd** は、コマンドラインユーティリティーの **chronyc** を使用して監視と管理を行います。このユーティリティーは、**chronyd** の現在の状態に対してクエリーを実行し、その設定を変更する多数のコマンド入力を可能にするコマンドプロンプトを提供します。デフォルトでは、**chronyd** は **chronyc** のローカルインスタンスのコマンドのみを受け付けますが、リモートホストから監視コマンドを受け付けるように設定することも可能です。リモートアクセスは制限する必要があります。

13.1.2. chronyc を使用した chronyd の制御

chronyc コマンドラインユーティリティーを使用して **chronyd** を制御できます。

手順

1. 対話モードでコマンドラインユーティリティー **chronyc** を使用して、**chronyd** のローカルインスタンスを変更するには、**root** で以下のコマンドを実行します。

```
# chronyc
```

制限されているコマンドを使用する場合は、**root** で **chronyc** を実行する必要があります。

以下のように、**chronyc** コマンドプロンプトが表示されます。

```
chronyc>
```

2. コマンドのリストを表示するには、**help** と入力します。
3. 以下のように、コマンドと合わせて呼び出した場合には、非対話的なコマンドモードでユーティリティーを呼び出すこともできます。

```
chronyc command
```

注記

chronyc を使用して変更した内容は永続的ではなく、**chronyd** を再起動すると元に戻ります。永続的に変更する場合は、**/etc/chrony.conf** を変更してください。

13.2. CHRONY の使用

次のセクションでは、**chronyd** のインストール、起動、停止の方法や、**chrony** が同期しているかどうかを確認する方法を説明します。また、システムクロックを手動で調整する方法も説明されています。

13.2.1. chrony の管理

以下の手順では、**chronyd** のインストール、起動、停止、およびステータスの確認方法を説明します。

手順

1. Red Hat Enterprise Linux では、**chrony** スイートがデフォルトでインストールされます。インストールされていることを確認するには、**root** で以下のコマンドを実行します。

```
# dnf install chrony
```

chrony デーモンのデフォルトの場所は、**/usr/sbin/chronyd** です。このコマンドラインユーティリティーは **/usr/bin/chronyc** にインストールされます。

2. **chronyd** のステータスを確認するには、以下のコマンドを実行します。

```
$ 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. **chronyd** を開始するには、**root** で以下のコマンドを実行します。

```
# systemctl start chronyd
```

システムの起動時に **chronyd** を自動的に起動するように設定するには、**root** で以下のコマンドを実行します。

```
# systemctl enable chronyd
```

4. **chronyd** を停止するには、**root** で以下のコマンドを実行します。

```
# systemctl stop chronyd
```

システムの起動時に **chronyd** を自動的に起動しないように設定するには、**root** で以下のコマンドを実行します。

```
# systemctl disable chronyd
```

13.2.2. chrony の同期確認

以下の手順では、**tracking** コマンド、**sources** コマンド、および **sourcestats** コマンドを使用して、**chrony** が同期されているかどうかを確認する方法を説明します。

手順

1. **chrony** の追跡を確認するには、以下のコマンドを実行します。

```
$ chronyc tracking
Reference ID    : CB00710F (ntp-server.example.net)
Stratum         : 3
Ref time (UTC)  : Fri Jan 27 09:49:17 2017
System time     :  0.000006523 seconds slow of NTP time
Last offset     : -0.000006747 seconds
RMS offset      : 0.000035822 seconds
Frequency       : 3.225 ppm slow
Residual freq   : 0.000 ppm
Skew            : 0.129 ppm
Root delay      : 0.013639022 seconds
Root dispersion : 0.001100737 seconds
Update interval : 64.2 seconds
Leap status     : Normal
```

2. sources コマンドは、**chronyd** がアクセスしている現在の時間ソースの情報を表示します。**chrony** ソースを確認するには、以下のコマンドを実行します。

```
$ 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
```

オプションの **-v** 引数を指定すると、より詳細な情報を出力できます。この例では、余分なキャプション行は、コラムの意味を説明するものとして表示されます。

3. **sourcestats** コマンドは、**chronyd** が現在調べている各ソースに関するドリフト量とオフセット推定プロセスの情報を表示します。**chrony** ソースの統計情報を確認するには、以下のコマンドを実行します。

```
$ 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
```

任意の引数 **-v** (verbose (詳細) の意) を指定できます。この例では、余分なキャプション行は、コラムの意味を説明するものとして表示されます。

関連情報

- **chronyc(1)** の man ページ

13.2.3. システムクロックの手動調整

以下の手順では、システムクロックを手動で調整する方法を説明します。

手順

1. システムクロックを徐々に調整していく (slew) のを止め、一度に修正 (step) するには、**root** で以下のコマンドを実行します。

```
# chronyc makestep
```

rtcfile ディレクティブを使用している場合は、リアルタイムクロックを手動で調整しないでください。ランダムな調整を行うと、リアルタイムクロックがずれる変化量を測定する必要がある **chrony** に影響を与えます。

13.2.4. chrony ディスパッチャースクリプトの無効化

chrony ディスパッチャースクリプトは、NTP サーバーのオンラインとオフラインの状態を管理します。システム管理者は、ディスパッチャースクリプトを無効にして、**chronyd** がサーバーを常にポーリングし続けるようにすることができます。

システムで NetworkManager を有効にしてネットワーク設定を管理する場合、NetworkManager はインターフェイスの再設定中、操作の停止または開始中に **chrony** ディスパッチャースクリプトを実行します。ただし、NetworkManager の外部で特定のインターフェイスまたはルートを設定すると、次の状況が発生する可能性があります。

1. NTP サーバーへのルートが存在しない場合にディスパッチャースクリプトが実行され、NTP サーバーがオフライン状態に切り替わる可能性があります。

2. 後でルートを確立すると、デフォルトではスクリプトは再実行されず、NTP サーバーはオフライン状態のままになります。

chronyd が、個別に管理されたインターフェイスを持つ NTP サーバーと確実に同期できるようにするには、ディスパッチャースクリプトを無効にします。

前提条件

- システムに NetworkManager をインストールして有効にしました。
- Root アクセス

手順

1. **chrony** ディスパッチャースクリプトを無効にするには、**/dev/null** へのシンボリックリンクを作成します。

```
# ln -s /dev/null /etc/NetworkManager/dispatcher.d/20-chrony-onoffline
```

注記

この変更後、NetworkManager はディスパッチャースクリプトを実行できなくなり、NTP サーバーは常にオンライン状態のままになります。

13.2.5. 孤立したネットワークでのシステムにおける chrony の設定

インターネットに接続されていないネットワークの場合、1 台のコンピューターがプライマリータイムサーバーとして選択されます。他のコンピューターは、サーバーの直接のクライアント、またはクライアントのクライアントです。サーバーでは、ドリフトファイルは、システムクロックのドリフトの平均率を使用して手動で設定します。サーバーを再起動すると、周囲のシステムから時間を取得し、システムクロックを設定するために平均値を計算します。その後、drift ファイルに基づいて調整の適用を再開します。drift ファイルは、**settime** コマンドが使用されたときに自動的に更新されます。

以下の手順では、分離ネットワークで asystem に **chrony** を設定する方法を説明します。

手順

1. マスターに選ばれたシステムで、**root** でテキストエディターを実行し、以下のように **/etc/chrony.conf** を実行します。

```
driftfile /var/lib/chrony/drift
commandkey 1
keyfile /etc/chrony.keys
initstepslew 10 client1 client3 client6
local stratum 8
manual
allow 192.0.2.0/24
```

192.0.2.0/24 は、クライアントが接続できるネットワークアドレスまたはサブネットアドレスです。詳細は **chrony.conf (7)** man ページを参照してください。

2. サーバーのダイレクトクライアントに選ばれたシステムで、**root** でテキストエディターを実行し、以下のように **/etc/chrony.conf** を実行します。

```
server ntp1.example.net
driftfile /var/lib/chrony/drift
logdir /var/log/chrony
log measurements statistics tracking
keyfile /etc/chrony.keys
commandkey 24
```

```
local stratum 10
initstepslew 20 ntp1.example.net
allow 192.0.2.123
```

192.0.2.123 はサーバーのアドレスで、**ntp1.example.net** はサーバーのホスト名です。この設定のクライアントは、再起動するとサーバーと再同期します。

サーバーの直接のクライアントにはならないクライアントシステムの **/etc/chrony.conf** ファイルでは、**local** ディレクティブおよび **allow** ディレクティブが省略される以外は、同じになるべきです。

孤立したネットワークでは、ローカルの参照モードを有効にする **local** ディレクティブも使用できます。これにより、**NTP** サーバーとして動作している **chronyd** は、サーバーが一度も同期されていなかったり、クロックの最終更新から長い時間が経過している場合でも、リアルタイムに同期してるように見えます。

複数のサーバーをポーリングしているクライアントを混同することなく、ネットワーク上の複数のサーバーに同じローカル設定を使用し、互いを同期させるには、Orphan モードを有効にする **local** ディレクティブの **orphan** オプションを使用します。各サーバーは、他のすべてのサーバーを **local** でポーリングするように設定する必要があります。これにより、最小の参照 ID を持つサーバーでのみローカル参照が有効になり、他のサーバーはそれに同期します。サーバーが失敗すると別のサーバーが引き継ぎます。

13.2.6. リモート監視アクセスの設定

chronyc は、以下の 2 つの方法で **chronyd** にアクセスします。

- インターネットプロトコル (IPv4 または IPv6)
- Unix ドメインソケット (ユーザー **root** または **chrony** がローカルにアクセス可能)

デフォルトでは、**chronyc** は、Unix ドメインソケットに接続します。デフォルトのパスは **/var/run/chrony/chronyd.sock** です。この接続に失敗すると (たとえば非特権ユーザーで **chronyc** を実行していると失敗する可能性があります)、**chronyc** は 127.0.0.1 への接続を試み、その後 ::1 への接続を試みます。

chronyd の動作に影響しない次の監視コマンドのみが、ネットワークに許可されています。

- activity
- manual list
- rtcdata
- smoothing
- sources
- sourcestats
- tracking
- waitsync

chronyd がこのコマンドを受け取るホスト郡は、**chronyd** の設定ファイルにある **cmdallow** ディレクティブ、または **chronyc** の **cmdallow** コマンドで設定できます。デフォルトでは、このコマンドが許可されるのは、ローカルホスト (127.0.0.1 または ::1) のものだけになります。

その他のコマンドはすべて、Unix ドメインソケットのみを介して許可されます。ネットワーク上で送信されると、たとえローカルホストであっても、**chronyd** は **Not authorised** エラーを返します。

以下の手順では、**chronyc** を使用して chronyd にリモートでアクセスする方法を説明します。

手順

1. 以下を **/etc/chrony.conf** ファイルに追加すると、IPv4 と IPv6 の両方のアドレスからアクセスが可能になります。

```
bindcmdaddress 0.0.0.0
```

または

```
bindcmdaddress ::
```

2. **cmdallow** ディレクティブを使用すると、リモート IP アドレス、ネットワーク、またはサブネットからのコマンドが許可されます。
/etc/chrony.conf ファイルに以下の内容を追加します。

```
cmdallow 192.168.1.0/24
```

3. ファイアウォールでポート 323 を開き、リモートシステムから接続します。

```
# firewall-cmd --zone=public --add-port=323/udp
```

必要に応じて、**--permanent** オプションを使用してポート 323 を永続的に開くことができます。

```
# firewall-cmd --permanent --zone=public --add-port=323/udp
```

4. ポート 323 を永続的に開く場合は、ファイアウォール設定を再読み込みします。

```
# firewall-cmd --reload
```

関連情報

- **chrony.conf(5)** の man ページ

13.2.7. RHEL システムロールを使用した時刻同期の管理

timesync ロールを使用して、複数のターゲットマシンで時刻同期を管理できます。**timesync** ロールは、NTP または PTP 実装をインストールして、NTP または PTP クライアントとして動作してシステムクロックと同期するように設定します。

警告

timesync ロールは、マネージドホストで指定または検出されたプロバイダーサービスの設定を置き換えます。以前の設定は、ロール変数で指定されていなくても失われます。**timesync_ntp_provider** 変数が定義されていない場合は、プロバイダーの唯一の設定が適用されます。

以下の例は、サーバーにプールが 1 つしかない場合に、**timesync** ロールを適用する方法を示しています。

例13.1 サーバーの 1 つのプールに、timesync ロールを適用する Playbook の例

```
---
- hosts: timesync-test
  vars:
    timesync_ntp_servers:
      - hostname: 2.rhel.pool.ntp.org
        pool: yes
        iburst: yes
  roles:
    - rhel-system-roles.timesync
```

timesync ロール変数の詳細は、**rhel-system-roles** パッケージをインストールし、**/usr/share/doc/rhel-system-roles/timesync** ディレクトリーの **README.md** または **README.html** ファイルを参照してください。

関連情報

- RHEL システムロールを使用するためのコントロールノードと管理対象ノードの準備

13.2.8. 関連情報

- **chronyc(1)** の man ページ
- **chronyd(8)** の man ページ
- よくある質問

13.3. ハードウェアのタイムスタンプを使用した CHRONY

ハードウェアのタイムスタンプは、一部の Network Interface Controller (NIC) でサポートされている機能です。着信パケットおよび送信パケットのタイムスタンプを正確に提供します。**NTP** タイムスタンプは通常、カーネルにより作成され、システムクロックを使用して **chronyd** が作成されます。ただし、ハードウェアのタイムスタンプが有効な場合、NIC は独自のクロックを使用して、パケットがリンク層または物理層に出入りするときにタイムスタンプを生成します。ハードウェアスタンプで **NTP** を使用すると、同期の精度を大幅に向上できます。最高精度を実現するには、**NTP** サーバーおよび **NTP** クライアントの両方が、ハードウェアのタイムスタンプを使用している必要があります。理想的な条件下では、サブマイクロ秒単位の精度を実現できるかもしれません。

ハードウェアのタイムスタンプを使用する時間同期の別のプロトコルには、**PTP** があります。

NTP とは異なり、**PTP** は、ネットワークスイッチおよびルーターの補助に依存しています。同期の精度を最高の状態にしたい場合は、**PTP** をサポートしているスイッチやルーターがあるネットワークで **PTP** を使用し、そのようなスイッチおよびルーターがないネットワークでは、**NTP** を使用することが推奨されます。

以下のセクションでは、次の方法を説明します。

- ハードウェアタイムスタンプのサポートの確認
- ハードウェアのタイムスタンプの有効化
- クライアントポーリング間隔の設定
- インターリーブモードの有効化
- 多数のクライアント向けのサーバー設定
- ハードウェアのタイムスタンプの確認
- PTP-NTP ブリッジの設定

13.3.1. ハードウェアタイムスタンプのサポートの確認

NTP を使用したハードウェアのタイムスタンプがインターフェイスでサポートされていることを確認するには、**ethtool -T** コマンドを実行します。**ethtool** が、**SOF_TIMESTAMPING_TX_HARDWARE** 機能、**SOF_TIMESTAMPING_TX_SOFTWARE** 機能、および **HWTSTAMP_FILTER_ALL** フィルターモードをリスト表示する場合は、**NTP** を使用して、ハードウェアのタイムスタンプにインターフェイスを使用できます。

例13.2 特定のインターフェイスにおけるハードウェアのタイムスタンプのサポートの確認

```
# ethtool -T eth0
```

出力:

```
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:
        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)
```

13.3.2. ハードウェアのタイムスタンプの有効化

ハードウェアのタイムスタンプを有効にするには、**/etc/chrony.conf** ファイルの **hwtimestamp** ディレクティブを使用します。ディレクティブは、個別のインターフェイスを指定できますが、ワイルドカード文字を使用して、ハードウェアのタイムスタンプをサポートするすべてのインターフェイスでハードウェアのタイムスタンプを有効にすることもできます。**linuxptp** パッケージの **ptp4l** などのアプリケーションが、ハードウェアのタイムスタンプを使用していない場合は、ワイルドカード仕様を使用してください。chrony 設定ファイルで、複数の **hwtimestamp** ディレクティブが使用されます。

例13.3 hwtimestamp のディレクティブを使用したハードウェアのタイムスタンプの有効化

```
hwtimestamp eth0
hwtimestamp eth1
hwtimestamp *
```

13.3.3. クライアントポーリング間隔の設定

インターネット上のサーバーのポーリング間隔は、デフォルトの範囲である 64 秒から 1024 秒が推奨されています。ローカルサーバーおよびハードウェアのタイムスタンプでは、システムクロックのオフセットを最小限にとどめるため、ポーリング間隔は短く設定する必要があります。

/etc/chrony.conf における以下のディレクティブは、1 秒のポーリング間隔を使用してローカルの **NTP** サーバーを指定します。

```
server ntp.local minpoll 0 maxpoll 0
```

13.3.4. インターリーブモードの有効化

ハードウェアの **NTP** アプライアンスではなく、**chrony** など、ソフトウェアの **NTP** 実装を実行する汎用コンピューターの **NTP** サーバーは、パケット送信後にのみハードウェア送信タイムスタンプを取得します。この動作により、サーバーは、対応するパケットのタイムスタンプを保存できません。**NTP** クライアントが、送信後に生成された送信タイムスタンプを受け取るようにするには、**/etc/chrony.conf** のサーバーディレクティブに **xleave** オプションを追加し、クライアントが **NTP** インターリーブモードを使用するように設定します。

```
server ntp.local minpoll 0 maxpoll 0 xleave
```

13.3.5. 多数のクライアント向けのサーバーの設定

デフォルトのサーバー設定では、最多で数千のクライアントが同時にインターリーブモードを使用できます。さらに多くのクライアント向けにサーバーを設定するには、**/etc/chrony.conf** の **clientloglimit** ディレクティブを増やします。このディレクティブは、サーバーでクライアントのアクセスログに割り当てられるメモリーの最大サイズを指定します。

```
clientloglimit 100000000
```

13.3.6. ハードウェアのタイムスタンプの確認

インターフェイスがハードウェアのタイムスタンプを有効にできたことを確認するには、システムログを確認してください。ログには、**chronyd** からの各インターフェイス向けメッセージに、有効にしたハードウェアのタイムスタンプが追記されているはずです。

例13.4 ハードウェアのタイムスタンプが有効になったインターフェイスのログメッセージ

```
chronyd[4081]: Enabled HW timestamping on eth0
chronyd[4081]: Enabled HW timestamping on eth1
```

chronyd が、**NTP** クライアントまたはピアとして設定されている場合は、**chronyc ntpdata** コマンドにより、送信先と受信先のタイムスタンプおよびインターリーブモードを、各 **NTP** ソースに報告できます。

例13.5 各 NTP ソースの送信先および受信先のタイムスタンプおよびインターリーブモードの報告

```
# chronyc ntpdata
```

出力:

```
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      : 0.000005396 seconds
Peer dispersion : 0.000002329 seconds
Response time   : 0.000152073 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
```

例13.6 NTP 測定の安定性の報告

```
# chronyc sourcestats
```

ハードウェアのタイムスタンプを有効にすると、**NTP** 測定の安定性は、通常のロードにおいて数十ナノ秒または数百ナノ秒となります。この安定性は、**chronyc sourcestats** コマンドの出力の **Std Dev** 列に報告されます。

出力:

```
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
```

13.3.7. PTP-NTP ブリッジの設定

非常に精度が高い **PTP** (Precision Time Protocol) のプライマリータイムセーバーが、 **PTP** サポートのあるスイッチまたはルーターを持たないネットワークで利用可能な場合、コンピューターは、**PTP** スレーブおよび stratum-1 の **NTP** サーバーとしての操作に専念する可能性があります。このようなコンピューターには、2 つ以上のネットワークインターフェイスが必要であり、プライマリータイムサーバーの近くに配置するか、プライマリータイムサーバーに直接接続する必要があります。これにより、ネットワークで非常に精度の高い同期が確実に実行されます。

1 つのインターフェイスを使用して、**PTP** でシステムクロックを同期するように、**linuxptp** パッケージの **ptp4l** プログラムおよび **phc2sys** プログラムを設定します。

chronyd を設定して、その他のインターフェイスを使用してシステム時間を提供するには、以下を行います。

例13.7 その他のインターフェイスを使用してシステム時間を提供するように chronyd を設定

```
bindaddress 203.0.113.74
hwtimestamp eth1
local stratum 1
```

13.4. CHRONY における NETWORK TIME SECURITY (NTS) の概要

Network Time Security (NTS) は、大規模なクライアントを拡張するように設計された Network Time Protocol (NTP) の認証メカニズムです。これは、クライアントマシンへの移動時に、サーバーマシンから受信したパケットが変更されていないことを確認します。NTS (Network Time Security) には、サーバーとそのクライアント間で使用される暗号鍵を自動的に作成する NTS-KE (Key Establishment) プロトコルが含まれます。

警告

NTS は、FIPS および OSPP プロファイルと互換性がありません。FIPS および OSPP プロファイルを有効にすると、NTS で設定された **chronyd** が致命的なメッセージを表示して中断する可能性があります。**GNUTLS_FORCE_FIPS_MODE=0** を **/etc/sysconfig/chronyd** ファイルに追加することで、**chronyd** サービスの OSPP プロファイルと FIPS モードを無効にできます。

13.4.1. クライアント設定ファイルでの Network Time Security (NTS) の有効化

デフォルトでは、Network Time Security (NTS) は有効になっていません。**/etc/chrony.conf** では、NTS を有効にできます。これを行うには、以下の手順を実行します。

前提条件

- NTS に対応するサーバー

手順

クライアント設定ファイル

1. 推奨される **iburst** オプションのほかに、**nts** オプションを使用してサーバーを指定します。

```
For example:
server time.example.com iburst nts
server nts.netnod.se iburst nts
server ptbtime1.ptb.de iburst nts
```

2. システムの起動時に Network Time Security-Key Establishment (NTS-KE) セッションが繰り返されないようにするには、次の行 (がない場合) を **chrony.conf** に追加します。

```
ntsdumpdir /var/lib/chrony
```

3. **DHCP** によって提供されるネットワークタイムプロトコル (NTP) サーバーとの Synchronization を無効にするには、**chrony.conf** に次の行がある場合は、コメントアウトするか削除します。

```
sourcedir /run/chrony-dhcp
```

4. 変更を保存します。
5. **chronyd** を再起動します。

```
systemctl restart chronyd
```

検証

- **NTS** キーが正常に確立されたかどうかを確認します。

```
# chronyc -N authdata
```

```
Name/IP address  Mode KeyID Type KLen Last Atmp  NAK Cook CLen
==============================================================
time.example.com  NTS     1   15  256  33m    0    0    8  100
nts.sth1.ntp.se   NTS     1   15  256  33m    0    0    8  100
nts.sth2.ntp.se   NTS     1   15  256  33m    0    0    8  100
```

KeyID、**Type**、および **KLen** には、ゼロ以外の値を指定する必要があります。この値が 0 になっていない場合は、システムログで **chronyd** からのエラーメッセージを確認します。

- クライアントが NTP 測定を行っていることを確認します。

```
# chronyc -N sources

MS Name/IP address Stratum Poll Reach LastRx Last sample
===============================================================
time.example.com  3       6   377    45   +355us[ +375us] +/-   11ms
nts.sth1.ntp.se    1       6   377    44   +237us[ +237us] +/-   23ms
nts.sth2.ntp.se    1       6   377    44   -170us[ -170us] +/-   22ms
```

Reach 列の値はゼロ以外にする必要があります。理想的には 377 です。この値が 377 になることがめったにないか、377 に到達しない場合は、NTP の要求または応答がネットワークで失われていることを示しています。

関連情報

- **chrony.conf(5)** の man ページ

13.4.2. サーバーで NTS (Network Time Security) の有効化

独自の Network Time Protocol (NTP) サーバーを実行している場合は、サーバーの Network Time Security (NTS) サポートを有効にして、クライアントの同期を容易にし、安全に行うことができます。

NTP サーバーがその他のサーバーのクライアントである (Stratum 1 サーバーではない) 場合は、同期に NTS または対称鍵を使用する必要があります。

前提条件

- **PEM** 形式のサーバー秘密鍵
- **PEM** 形式で必要な中間証明書を持つサーバー証明書

手順

1. **chrony.conf** で秘密鍵と証明書ファイルを指定します。以下に例を示します。

```
ntsserverkey /etc/pki/tls/private/<ntp-server.example.net>.key
ntsservercert /etc/pki/tls/certs/<ntp-server.example.net>.crt
```

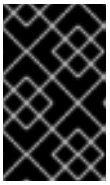
2. グループの所有権を設定し、鍵と証明書ファイルの両方が chrony システムユーザーにより読み取り可能であることを確認します。以下に例を示します。

```
# chown :chrony /etc/pki/tls//<ntp-server.example.net>.
```

3. **ntsdumpdir /var/lib/chrony** ディレクティブが **chrony.conf** に存在することを確認します。

4. **chronyd** を再起動します。

```
# systemctl restart chronyd
```



重要

サーバーにファイアウォールがある場合は、NTP 用の **UDP 123** ポートと **TCP 4460** ポート、および NTS-KE (Network Time Security-Key Establishment) の両方を許可する必要があります。

検証

- 次のコマンドを使用して、クライアントマシンからクイックテストを実行します。

```
$ chronyd -Q -t 3 'server

ntp-server.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
```

System clock wrong メッセージは、NTP サーバーが NTS-KE 接続を受け入れ、NTS で保護されている NTP メッセージで応答していることを示しています。

- サーバーで監視されている NTS-KE 接続と認証された NTP パケットを確認します。

```
# 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
```

NTS-KE connections accepted および **Authenticated NTP packets** の値がゼロ以外の値の場合は、少なくとも 1 台のクライアントが NTS-KE ポートに接続し、認証された NTP リクエストを送信できたことを意味します。

第14章 システムの復旧および復元

Red Hat Enterprise Linux では、既存のバックアップを使用してシステムを復旧および復元するために、ReaR (Relax-and-Recover) ユーティリティーが同梱されています。

このユーティリティーは、障害復旧ソリューションとして、またシステム移行にも使用できます。

このユーティリティーを使用すると、以下のタスクを実行できます。

- イメージを使用して、起動可能なイメージを作成し、既存のバックアップからシステムを復元する
- オリジナルのストレージレイアウトを複製する
- ユーザーおよびシステムファイルを復元する
- システムを別のハードウェアに復元する

また、障害復旧の場合は、特定のバックアップソフトウェアを ReaR に統合することもできます。

ReaR 設定の概要手順は以下のとおりです。

1. ReaR をインストールします。
2. ReaR 設定ファイルを変更して、バックアップ手法の詳細を追加します。
3. レスキューシステムを作成します。
4. バックアップファイルを生成します。

14.1. REAR の設定

以下の手順を使用して、Relax-and-Recover (ReaR) ユーティリティーを使用するパッケージのインストール、レスキューシステムの作成、バックアップの設定および生成を行います。

前提条件

- バックアップ復元計画をもとに、必要な設定ができている。
  NETFS バックアップメソッド (ReaR に完全に統合され、組み込まれたメソッド) を使用できることに注意してください。

手順

1. 次のコマンドを実行して ReaR ユーティリティーをインストールします。

```
# dnf install rear
```

2. 以下の例のように、任意のエディターで ReaR 設定ファイルを変更します。

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

3. バックアップ設定の詳細を **/etc/rear/local.conf** に追加します。たとえば、**NETFS** バックアップメソッドの場合は、以下の行を追加します。

```
BACKUP=NETFS
BACKUP_URL=backup.location
```

backup.location は、バックアップ先の URL に置き換えます。

4. 新規バックアップの作成時に以前のバックアップアーカイブを維持するように RaaR 設定を行うには、以下の行を設定ファイルに追加します。

```
NETFS_KEEP_OLD_BACKUP_COPY=y
```

5. 増分バックアップ (実行するたびに変更されたファイルのみがバックアップされる) を設定する場合は、以下の行を追加します。

```
BACKUP_TYPE=incremental
```

6. レスキューシステムを作成します。

```
# rear mkrescue
```

7. 復元計画に従ってバックアップを作成します。たとえば、**NETFS** バックアップメソッドの場合は、以下のコマンドを実行します。

```
# rear mkbackuponly
```

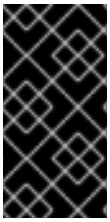
または、以下のコマンドを実行すると、1 つの手順でレスキューシステムとバックアップを作成できます。

```
# rear mkbackup
```

このコマンドは、**rear mkrescue** コマンドと **rear mkbackuponly** コマンドの機能を組み合わせたものです。

14.2. 64 ビット IBM Z アーキテクチャーで REAR レスキューイメージの使用

Basic Relax and Recover (ReaR) 機能が、64 ビットの IBM Z アーキテクチャーでテクノロジープレビューとして利用できるようになりました。IBM Z では、z/VM 環境でのみ ReaR レスキューイメージを作成できます。論理パーティション (LPAR) のバックアップおよび復元はテストされていません。



重要

64 ビット IBM Z アーキテクチャーでの ReaR は、**rear** パッケージのバージョン 2.6-17.el9 以降でのみサポートされます。以前のバージョンは、テクノロジープレビュー機能としてのみ利用できます。Red Hat のテクノロジープレビュー機能のサポート範囲に関する詳細は、テクノロジープレビュー機能のサポート範囲 を参照してください。

現在利用できる出力方法は、Initial Program Load (IPL) のみです。IPL は、**zIPL** ブートローダーで使用できるカーネルと初期 RAM ディスク (initrd) を生成します。

前提条件

- ReaR がインストールされている。

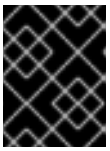
- ReaR をインストールするには、**dnf install rear** コマンドを実行します。

手順

以下の変数を**/etc/rear/local.conf**に追加し、64 ビット IBM Z アーキテクチャーでレスキューイメージを生成するように ReaR を設定します。

1. **IPL** アウトプットメソッドを設定するには、**OUTPUT=IPL** を追加します。
2. バックアップメソッドとバックアップ先を設定するには、**BACKUP** 変数および **BACKUP_URL** 変数を追加します。以下に例を示します。

```
BACKUP=NETFS

BACKUP_URL=nfs://<nfsserver name>/<share path>
```



重要

ローカルバックアップストレージは、現在、64 ビットの IBM Z アーキテクチャーではサポートされていません。

3. 必要に応じて、**OUTPUT_URL** を設定して、カーネルファイルおよび **initrd** ファイルを保存することもできます。初期設定では、**OUTPUT_URL** は **BACKUP_URL** に合わせて配置されています。
4. バックアップとレスキューのイメージの作成を実行するには、次のコマンドを実行します。

```
# rear mkbackup
```

5. これにより、**BACKUP_URL** 変数または **OUTPUT_URL** (設定されている場合) 変数で指定された場所にカーネルファイルと initrd ファイルが作成され、指定されたバックアップメソッドを使用してバックアップが作成されます。
6. システムを復元するには、手順 3 で作成した ReaR カーネルファイルおよび initrd ファイルを使用し、**zipl** ブートローダー、カーネル、および **initrd** で準備した DASD (Direct Attached Storage Device) または FCP (Fibre Channel Protocol) 接続の SCSI デバイスから起動します。詳細は準備した DASD の使用を参照してください。
7. レスキューカーネルと **initrd** が起動すると、ReaR レスキュー環境が起動します。システムの復元を続行します。

警告

現在、レスキュープロセスは、システムに接続したすべての DASD (Direct Attached Storage Devices) を再フォーマットします。システムのストレージデバイスに貴重なデータが存在する場合は、システムの復旧を行わないでください。これには、レスキュー環境で起動するのに使用された zipl ブートローダー、ReaR カーネル、および initrd で準備されたデバイスも含まれます。必ずコピーを保管してください。

関連情報

- z/VM へのインストール
- 設定済み DASD の使用