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: 2025-08-04

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

Legal Notice

Abstract

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。

Table of Contents

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

NetworkManager は、ホストにインストールされている各イーサネットアダプターの接続プロファイルを作成します。デフォルトでは、このプロファイルは IPv4 接続と IPv6 接続の両方に DHCP を使用します。次の場合は、この自動作成されたプロファイルを変更するか、新しいプロファイルを追加してください。

- ネットワークに、静的 IP アドレス設定などのカスタム設定が必要な場合
- ホストが異なるネットワーク間をローミングするため、複数のプロファイルが必要な場合

Red Hat Enterprise Linux では、管理者はさまざまな方法でイーサネット接続を設定できます。以下に例を示します。

- nmcli を使用して、コマンドラインで接続を設定する。
- nmtui 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を設定する。
- GNOME 設定メニューまたは nm-connection-editor アプリケーションを使用して、グラフィカルインターフェイスで接続を設定する。
- nmstatectl を使用して、Nmstate API を介して接続を設定する。
- RHEL システムロールを使用して、1 つまたは複数のホストで接続の設定を自動化する。

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。
3. **ON/OFF** スイッチを切り替えて、選択したインターフェイスを有効または無効にします。インターフェイスを手動で追加または削除することはできません。
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、チーム (非推奨)、ボンド、ブリッジ、または VLAN のいずれかです。
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。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。

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。

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。

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。

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

- RHEL の自動インストール
- ネットワークデバイスの命名標準の詳細は、**ネットワークの設定および管理** を参照してください。

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、次のように実行します。

```
# 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、次のように実行します。

```
# 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、次のように実行します。

```
# 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、次のように実行します。

```
# 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、次のように実行します。

```
# 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::fffe 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行し、不具合のあるケーブルとネットワークインターフェイスカードを交換します。
- 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。この問題の詳細と回避方法は、Red Hat ナレッジベースソリューション NetworkManager duplicates a connection after restart of NetworkManager service を参照してください。

関連情報

- システム上の **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。
- 前の画面に戻るには、**ESC** を使用します。

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。

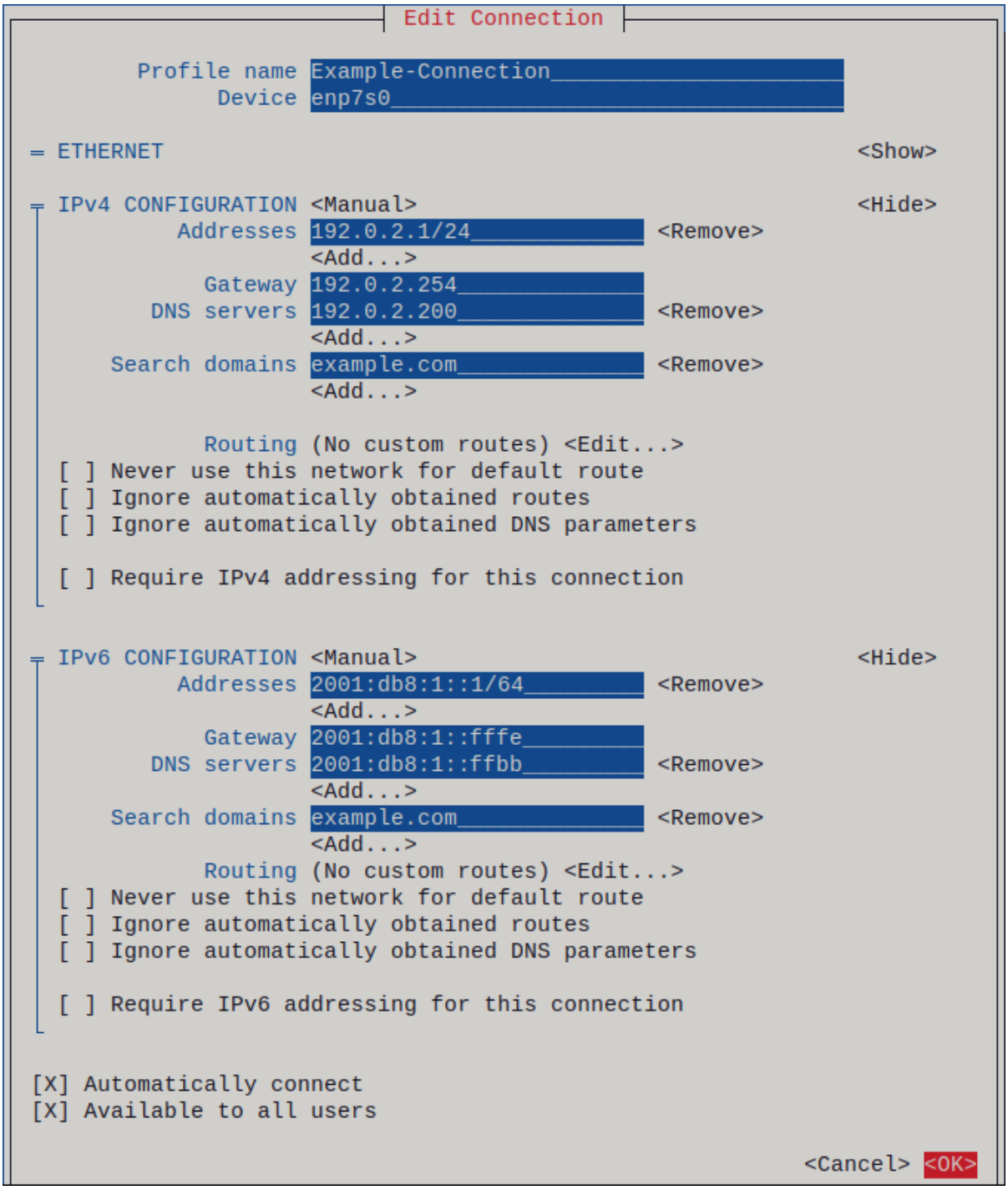
     ii. **Addresses** の横にある **Add** を押して、IP アドレスとサブネットマスクを Classless Inter-Domain Routing (CIDR) 形式で入力します。
     サブネットマスクを指定しない場合、NetworkManager は IPv4 アドレスに **/32** サブネットマスクを設定し、IPv6 アドレスに **/64** サブネットマスクを設定します。

     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::fffe 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行し、不具合のあるケーブルとネットワークインターフェイスカードを交換します。
- 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。この問題の詳細と回避方法は、Red Hat ナレッジベースソリューション NetworkManager duplicates a connection after restart of NetworkManager service 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. NETWORK RHEL システムロールとインターフェイス名を使用した動的 IP アドレスでのイーサネット接続の設定

Red Hat Enterprise Linux ホストをイーサネットネットワークに接続するには、ネットワークデバイスの NetworkManager 接続プロファイルを作成します。Ansible と **network** RHEL システムロールを使用すると、このプロセスを自動化し、Playbook で定義されたホスト上の接続プロファイルをリモートで設定できます。

network RHEL システムロールを使用すると、DHCP サーバーおよび IPv6 ステートレスアドレス自動設定 (SLAAC) から IP アドレス、ゲートウェイ、および DNS 設定を取得するイーサネット接続を設定できます。このロールを使用すると、指定のインターフェイス名に接続プロファイルを割り当てることができます。

前提条件

- コントロールノードと管理対象ノードの準備が完了している。
- 管理対象ノードで Playbook を実行できるユーザーとしてコントロールノードにログインしている。
- 管理対象ノードへの接続に使用するアカウントに、そのノードに対する **sudo** 権限がある。
- サーバーの構成に物理または仮想イーサネットデバイスが存在する。
- ネットワーク内で DHCP サーバーと SLAAC が利用できる。
- 管理対象ノードが NetworkManager サービスを使用してネットワークを設定している。

手順

1. 次の内容を含む Playbook ファイル (例: ~/**playbook.yml**) を作成します。

```
---
- name: Configure the network
  hosts: managed-node-01.example.com
  tasks:
    - name: Ethernet connection profile with dynamic IP address settings
      ansible.builtin.include_role:
        name: redhat.rhel_system_roles.network
      vars:
        network_connections:
          - name: enp1s0
            interface_name: enp1s0
            type: ethernet
            autoconnect: yes
            ip:
              dhcp4: yes
              auto6: yes
            state: up
```

サンプル Playbook で指定されている設定は次のとおりです。

dhcp4: yes

DHCP、PPP、または同様のサービスからの自動 IPv4 アドレス割り当てを有効にします。

auto6: yes

IPv6 自動設定を有効にします。デフォルトでは、NetworkManager はルーター広告を使用します。ルーターが **managed** フラグを通知すると、NetworkManager は DHCPv6 サーバーに IPv6 アドレスと接頭辞を要求します。

Playbook で使用されるすべての変数の詳細は、コントロールノードの **/usr/share/ansible/roles/rhel-system-roles.network/README.md** ファイルを参照してください。

2. Playbook の構文を検証します。

```
$ ansible-playbook --syntax-check ~/playbook.yml
```

このコマンドは構文を検証するだけであり、有効だが不適切な設定から保護するものではないことに注意してください。

3. Playbook を実行します。

```
$ ansible-playbook ~/playbook.yml
```

検証

- 管理対象ノードの Ansible fact をクエリーし、インターフェイスが IP アドレスと DNS 設定を受信したことを確認します。

```
# ansible managed-node-01.example.com -m ansible.builtin.setup
...
        "ansible_default_ipv4": {
            "address": "192.0.2.1",
            "alias": "enp1s0",
            "broadcast": "192.0.2.255",
            "gateway": "192.0.2.254",
            "interface": "enp1s0",
            "macaddress": "52:54:00:17:b8:b6",
            "mtu": 1500,
            "netmask": "255.255.255.0",
            "network": "192.0.2.0",
            "prefix": "24",
            "type": "ether"
        },
        "ansible_default_ipv6": {
            "address": "2001:db8:1::1",
            "gateway": "2001:db8:1::fffe",
            "interface": "enp1s0",
            "macaddress": "52:54:00:17:b8:b6",
            "mtu": 1500,
            "prefix": "64",
            "scope": "global",
            "type": "ether"
        },
        ...
        "ansible_dns": {
            "nameservers": [
                "192.0.2.1",
                "2001:db8:1::ffbb"
```

```
            ],
            "search": [
                "example.com"
            ]
        },
...
```

関連情報

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

1.5. 関連情報

- ネットワークの設定および管理

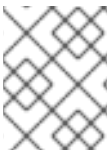
第2章 システム登録およびサブスクリプション管理

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。システムを登録していない場合、RHEL リポジトリーにアクセスすることはできません。セキュリティー、バグ修正などのソフトウェア更新をインストールすることはできません。Self-Support サブスクリプションをお持ちの場合でも、ナレッジベースにアクセスできます。一方、サブスクリプションがない場合は、多くのリソースを利用できません。サブスクリプションを購入し、Red Hat コンテンツ配信ネットワーク (CDN) を使用すると、次のものを追跡できます。

- 登録したシステム
- 登録したシステムにインストールされた製品
- 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. コマンドラインを使用したシステムの登録

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。システムを登録していない場合、RHEL リポジトリーにアクセスすることはできません。セキュリティー、バグ修正などのソフトウェア更新をインストールすることはできません。Self-Support サブスクリプションをお持ちの場合でも、ナレッジベースにアクセスできます。一方、サブスクリプションがない場合は、多くのリソースを利用できません。Red Hat アカウントの Red Hat Enterprise Linux サブスクリプションを有効化および管理するには、システムを登録する必要があります。



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。詳細は、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。

前提条件

- Red Hat Enterprise Linux システムの有効なサブスクリプションを持っている。

手順

- システムを登録してサブスクライブします。

```
# subscription-manager register
Registering to:                 subscription.rhsm.redhat.com:443/subscription
Username: <example_username>
Password: <example_password>
The system has been registered with ID: 37to907c-ece6-49ea-9174-20b87ajk9ee7
The registered system name is:          client1.example.com
```

このコマンドを実行すると、Red Hat カスタマーポータルアカウントのユーザー名とパスワードの入力を求められます。

登録プロセスが失敗した場合は、特定のプールにシステムを登録できます。詳細を確認するには、次の手順に進んでください。

  - サブスクリプションのプール ID を確認します。

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

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。

- **<example_pool_id>** を前のステップで確認したプール 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します。

```
# subscription-manager attach --pool=<example_pool_id>
```

検証

- Hybrid Cloud Console の **Inventory → Systems** でシステムを確認します。

関連情報

- Red Hat Subscription Management について
- Red Hat 製品を使用する際のワークフローの理解
- Hybrid Cloud Console でのサブスクリプションインベントリーの表示

2.2. WEB コンソールを使用したシステムの登録

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。システムを登録していない場合、RHEL リポジトリーにアクセスすることはできません。セキュリティー、バグ修正などのソフトウェア更新をインストールすることはできません。Self-Support サブスクリプションをお持ちの場合でも、ナレッジベースにアクセスできます。一方、サブスクリプションがない場合は、多くのリソースを利用できません。Red Hat Web コンソールでアカウント認証情報を使用して、新しくインストールした Red Hat Enterprise Linux を登録できます。

前提条件

- RHEL システムの有効なサブスクリプションを持っている。
- RHEL 9 Web コンソールがインストールされている。
- cockpit サービスが有効になっている。
- ユーザーアカウントが Web コンソールにログインできる。
  手順は、Web コンソールのインストールおよび有効化 を参照してください。

手順

1. ブラウザーで **https://<ip_address_or_hostname>:9090** を開き、Web コンソールにログインします。
2. **Overview** ページの **Health** フィールドで、**Not registered** の警告をクリックするか、メインメニューの **Subscriptions** をクリックして、サブスクリプション情報を含むページに移動します。
3. **Overview** フィールドで、**Register** をクリックします。
4. **Register system** ダイアログで、登録方法を選択します。

オプション: 組織の名前または 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 ID を追加する必要があります。組織 ID を取得する場合は、Red Hat のテクニカルアカウントマネージャーに確認してください。

5. システムを Red Hat Insights に接続しない場合は、**Insights** チェックボックスをオフにします。

6. **Register** をクリックします。

検証

- Hybrid Cloud Console でサブスクリプションの詳細を確認します。

2.3. GNOME デスクトップ環境でのシステムの登録

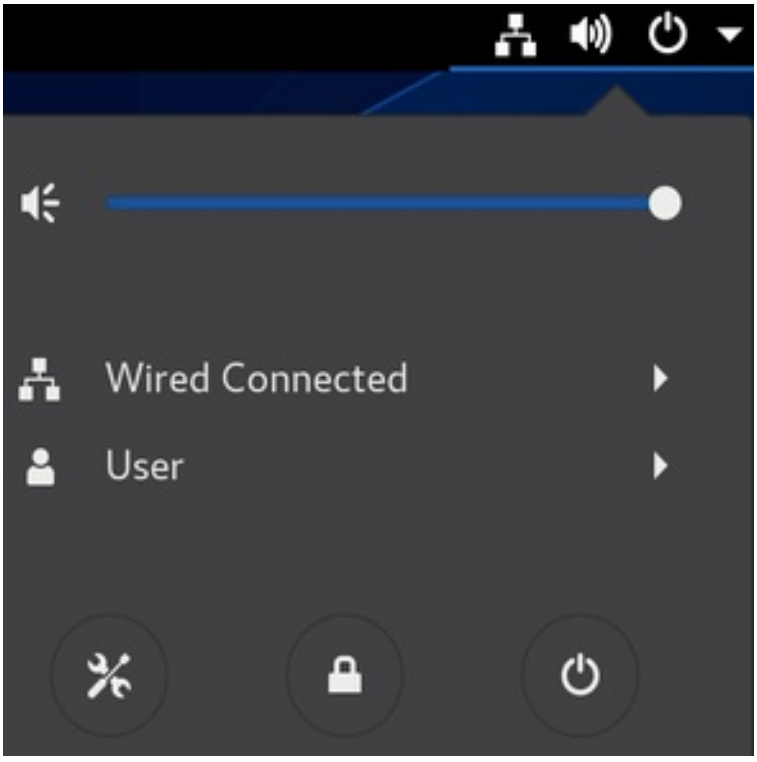
Red Hat Enterprise Linux オペレーティングシステムと、そこにインストールされている製品は、サブスクリプションの対象となります。システムを登録していない場合、RHEL リポジトリーにアクセスすることはできません。セキュリティー、バグ修正などのソフトウェア更新をインストールすることはできません。Self-Support サブスクリプションをお持ちの場合でも、ナレッジベースにアクセスできます。一方、サブスクリプションがない場合は、多くのリソースを利用できません。以下の手順に従って、システムを Red Hat アカウントに登録します。

前提条件

- Red Hat アカウント が作成されている。
- root ユーザーであり、GNOME デスクトップ環境にログインしている。詳細は、RHEL システムを Red Hat Subscription Manager に登録してサブスクライブする方法 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、**Settings** をクリックします。



2. **About → Subscription** に移動します。

3. Red Hat Satellite を通じてシステムを登録する場合は、以下を実行します。
    a. **Registration Server** セクションで、**Custom Address** 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. **Registration Type** セクションで、希望する登録方法を選択します。
5. **Registration Details** セクションに入力します。
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

問題のトラブルシューティングについてサポートが必要な場合は、Red Hat サポートにお問い合わせください。

手順

- Red Hat サポートの Web サイトにログインし、次のいずれかの方法を選択します。
  - サポートケースを作成する
  - Red Hat 専門スタッフとのライブチャットを開始する
  - 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

3.1. SOSREPORT ユーティリティーを使用してシステムに関する診断情報を収集し、サポートチケットに添付する

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
```

2. レポートを生成します。

```
# sosreport
```

   オプションで、レポートを自動的に **アップロード** してサポートケースに添付するには、コマンドに--upload オプションを渡します。これにはインターネットアクセスとカスタマーポータル認証情報が必要です。

3. オプション: レポートをサポートケースに手動で添付します。
   詳細は、Red Hat ナレッジベースソリューション How can I attach a file to a Red Hat support case? を参照してください。

関連情報

- What is an sosreport and how to create one in Red Hat Enterprise Linux? (Red Hat ナレッジベース)

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** パッケージから利用できます。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. 日付、時刻、およびタイムゾーン設定の手動設定

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. オプション: タイムゾーンをリスト表示します。

```
# timedatectl list-timezones

 Europe/Berlin
```

2. タイムゾーンを設定します。

```
# timedatectl set-timezone <time_zone>
```

3. 日付と時刻を設定します。

```
# timedatectl set-time <YYYY-mm-dd HH:MM:SS>
```

検証

1. 日付、時刻、タイムゾーンを表示します。

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

2. 詳細を表示するには、timedatectl コマンドを使用します。

```
# 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

- **date(1)** および **timedatectl(1)** man ページ

4.2. WEB コンソールを使用した時間の設定

RHEL Web コンソールでタイムゾーンを設定し、システム時間を Network Time Protocol (NTP) サーバーと同期できます。

前提条件

- RHEL 9 Web コンソールがインストールされている。
- cockpit サービスが有効になっている。
- ユーザーアカウントが Web コンソールにログインできる。
  手順は、Web コンソールのインストールおよび有効化 を参照してください。

手順

1. RHEL 9 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** で現在のシステム時間をクリックします。
3. **System time** をクリックします。
4. 必要に応じて、**システム時間の変更** ダイアログボックスで、タイムゾーンを変更します。
5. **Set Time** ドロップダウンメニューで、以下のいずれかを選択します。

   手動

   NTP サーバーなしで手動で時間を設定する必要がある場合は、このオプションを使用します。

   NTP サーバーの自動使用

   これはデフォルトのオプションで、既存の NTP サーバーと時間を自動同期します。

   特定の NTP サーバーの自動使用

   このオプションは、システムを特定の NTP サーバーと同期する必要がある場合に限り使用してください。サーバーの DNS 名または IP アドレスを指定します。
6. **Change** をクリックします。

検証

- **システム** タブに表示されるシステム時間を確認します。

関連情報

- Chrony スイートを使用した NTP の設定

4.3. 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

1. オプション: 現在のシステムロケール設定を表示します。

```
# localectl status
System Locale: LANG=en_US.UTF-8
 VC Keymap: de-nodeadkeys
   X11 Layout: de
  X11 Variant: nodeadkeys
```

2. 利用可能なシステムロケール設定をリスト表示します。

```
$ localectl list-locales
C.UTF-8
...
en_US.UTF-8
en_ZA.UTF-8
en_ZW.UTF-8
...
```

3. システムのロケール設定を更新します。

以下に例を示します。

+

```
# localectl set-locale LANG=en_US.UTF-8
```

注記

GNOME ターミナルは、UTF8 以外のシステムロケールをサポートしていません。詳細は、Red Hat ナレッジベースソリューション The gnome-terminal application fails to start when the system locale is set to non-UTF8 を参照してください。

関連情報

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

4.4. 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

1. 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
...
```

2. 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
...
```

3. 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)** man ページ

4.5. テキストコンソールモードでフォントサイズの変更

仮想コンソールでフォントサイズを変更できます。

手順

1. 現在使用されているフォントファイルを表示します。

```
# cat /etc/vconsole.conf

FONT="eurlatgr"
```

2. 利用可能なフォントファイルをリスト表示します。

```
# ls -1 /usr/lib/kbd/consolefonts/*.psfu.gz

/usr/lib/kbd/consolefonts/eurlatgr.psfu.gz
/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
```

お使いの文字セットとコードページをサポートするフォントファイルを選択します。

3. オプション: フォントファイルをテストするには、一時的に読み込みます。

```
# setfont LatArCyrHeb-16.psfu.gz
```

setfont ユーティリティーによってフォントファイルが直ちに適用されます。再起動するか別のフォントファイルを適用するまで、ターミナルで新しいフォントサイズが使用されます。

4. **/etc/vconsole.conf** で定義されたフォントファイルに戻すには、パラメーターなしで **setfont** と入力します。

5. **/etc/vconsole.conf** ファイルを編集し、**FONT** 変数を、RHEL が起動時にロードするフォントファイルに設定します。次に例を示します。

```
FONT=LatArCyrHeb-16
```

6. ホストを再起動します。

```
# reboot
```

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 や 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します。これにより、侵入者が接続から暗号化されていないパスワードを収集するのを防ぎます。

5.1. SSH 鍵ペアの生成

ローカルシステムで SSH 鍵ペアを生成し、生成された公開鍵を OpenSSH サーバーにコピーすることで、パスワードを入力せずに OpenSSH サーバーにログインできます。鍵を作成する各ユーザーは、この手順を実行する必要があります。

システムを再インストールした後も以前に生成した鍵ペアを保持するには、新しい鍵を作成する前に **~/.ssh/** ディレクトリーをバックアップします。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。

前提条件

- OpenSSH サーバーに鍵を使用して接続するユーザーとしてログインしている。
- OpenSSH サーバーが鍵ベースの認証を許可するように設定されている。

手順

1. ECDSA 鍵ペアを生成します。

```
$ ssh-keygen -t ecdsa
Generating public/private ecdsa key pair.
Enter file in which to save the key (/home/<username>/.ssh/id_ecdsa):
Enter passphrase (empty for no passphrase): <password>
Enter same passphrase again: <password>
Your identification has been saved in /home/<username>/.ssh/id_ecdsa.
Your public key has been saved in /home/<username>/.ssh/id_ecdsa.pub.
The key fingerprint is:
SHA256:Q/x+qms4j7PCQ0qFd09iZEFHA+SqwBKRNaU72oZfaCI
<username>@<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]-----+
```

パラメーターなしで **ssh-keygen** コマンドを使用して RSA 鍵ペアを生成することも、**ssh-keygen -t ed25519** コマンドを入力して Ed25519 鍵ペアを生成することもできます。Ed25519

アルゴリズムは FIPS-140 に準拠しておらず、FIPS モードでは OpenSSH は Ed25519 鍵で機能しないことに注意してください。

2. 公開鍵をリモートマシンにコピーします。

```
$ ssh-copy-id <username>@<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
<username>@<ssh-server-example.com>'s password:
…
Number of key(s) added: 1

Now try logging into the machine, with: "ssh '<username>@<ssh-server-example.com>'"
and check to make sure that only the key(s) you wanted were added.
```

<username>@<ssh-server-example.com> は、認証情報に置き換えます。

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。

検証

- 鍵ファイルを使用して OpenSSH サーバーにログインします。

```
$ ssh -o PreferredAuthentications=publickey <username>@<ssh-server-example.com>
```

関連情報

- システム上の **ssh-keygen(1)** および **ssh-copy-id(1)** man ページ

5.2. OPENSSH サーバーで鍵ベースの認証を唯一の方法として設定する

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。
- 鍵を使用して OpenSSH サーバーに接続できる。
  詳細は、SSH 鍵ペアの生成 セクションを参照してください。

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
```

3. 新しいデフォルトインストール以外のシステムでは、**PubkeyAuthentication** パラメーターが設定されていないか、**yes** に設定されていることを確認します。

4. **KbdInteractiveAuthentication** ディレクティブを **no** に設定します。
設定ファイル内では対応するエントリーがコメントアウトされていること、およびデフォルト値が **yes** であることに注意してください。

5. 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
```

6. 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。

7. **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

- システム上の **sshd_config(5)** および **setsebool(8)** man ページ

5.3. SSH-AGENT を使用した SSH 認証情報のキャッシュ

SSH 接続を開始するたびにパスフレーズを入力しなくても済むように、**ssh-agent** ユーティリティーを使用して、ログインセッションの SSH 秘密鍵をキャッシュできます。エージェントが実行中で、鍵のロックが解除されている場合、鍵のパスワードを再入力することなく、この鍵を使用して SSH サーバーにログインできます。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 デーモンが実行されており、ネットワーク経由でアクセスできる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 セクションを参照してください。

手順

1. セッションで **ssh-agent** を自動的に起動するためのコマンドを ~/**.bashrc** ファイルに追加します。

   a. 任意のテキストエディターで ~/**.bashrc** を開きます。次に例を示します。

   ```
   $ vi ~/.bashrc
   ```

   b. 以下の行をファイルに追加します。

   ```
   
   ```

```
eval $(ssh-agent)
```

c. 変更を保存し、エディターを終了します。

2. ~/**.ssh**/**config** ファイルに次の行を追加します。

```
AddKeysToAgent yes
```

セッションでこのオプションを使用して **ssh-agent** が起動されると、エージェントはホストに初めて接続するときにのみパスワードを要求します。

検証

- エージェントにキャッシュされた秘密鍵に対応する公開鍵を使用するホストにログインします。次に例を示します。

```
$ ssh <example.user>@<ssh-server@example.com>
```

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。

5.4. スマートカードに保存した SSH 鍵による認証

スマートカードに ECDSA 鍵と RSA 鍵を作成して保存し、そのスマートカードを使用して OpenSSH クライアントで認証することができます。スマートカード認証は、デフォルトのパスワード認証に代わるものです。

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
```

2. 公開鍵をリモートサーバーに転送します。**ssh-copy-id** コマンドを使用し、前の手順で作成した **keys.pub** ファイルを指定します。

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

3. ECDSA 鍵を使用して **<ssh-server-example.com>** に接続します。鍵を一意に参照する URI のサブセットのみを使用することもできます。次に例を示します。

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

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

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

4. オプション: **~/.ssh/config** ファイルで同じ URI 文字列を使用して、設定を永続的にすることができます。

```
$ cat ~/.ssh/config
IdentityFile "pkcs11:id=%01?module-path=/usr/lib64/pkcs11/opensc-pkcs11.so"
$ ssh <ssh-server-example.com>
Enter PIN for 'SSH key':
[ssh-server-example.com] $
```

ssh クライアントユーティリティーが、この URI とスマートカードの鍵を自動的に使用するようになります。

関連情報

- システム上の **p11-kit(8)**、**opensc.conf(5)**、**pcscd(8)**、**ssh(1)**、および **ssh-keygen(1)** man ページ

5.5. 関連情報

- システムの **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
- 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 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。

6.1. 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

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。

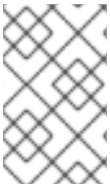
- カーネル
- ブートプロセスの初期段階
- 起動時および実行時のデーモンの標準出力とエラー
- **Syslog**
- **Rsyslog** サービス

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。

6.2. SYSLOG メッセージを保存するログファイル

/var/log ディレクトリーの次のログファイルには、**syslog** メッセージが保存されます。

- **/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



注記

上記のリストには一部のファイルのみが含まれています。**/var/log/** ディレクトリー内の実際のファイルリストは、このディレクトリーにログインするサービスとアプリケーションによって異なります。

6.3. 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処し、システムの他の部分と密接に統合されており、さまざまなロギングテクノロジーとログエントリーの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、コマンドラインからシステムジャーナル内のメッセージを表示できます。

表6.1 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6.2 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。
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6.3 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。

6.4. WEB コンソールでログの確認

RHEL Web コンソールでログへのアクセス、確認、およびフィルタリングの方法を説明します。

6.4.1. Web コンソールでログの確認

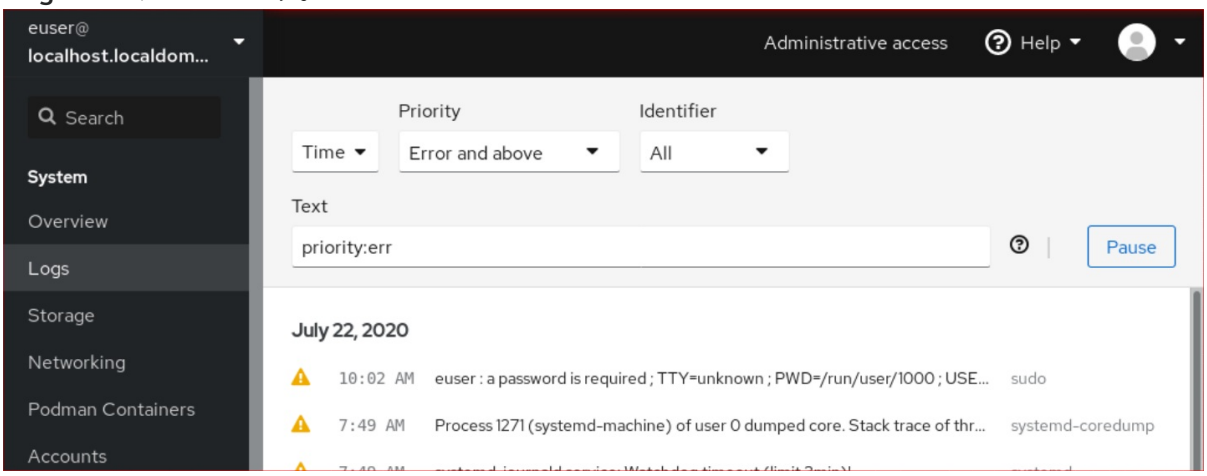
RHEL 9 Web コンソールのログセクションは、**journalctl** ユーティリティーの UI です。Web コンソールインターフェイスで、システムログにアクセスできます。

前提条件

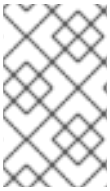
- RHEL 9 Web コンソールがインストールされている。
- cockpit サービスが有効になっている。
- ユーザーアカウントが Web コンソールにログインできる。
  手順は、Web コンソールのインストールおよび有効化 を参照してください。

手順

1. RHEL 9 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。



3. リストからログを確認するログエントリーをクリックして、ログエントリーの詳細を開きます。



注記

Pause ボタンを使用すると、新しいログエントリーが表示されないように一時停止できます。新しいログエントリーを再開すると、Web コンソールは、**Pause** ボタンを使用した後に報告されたすべてのログエントリーを読み込みます。

Priority 時間、優先順位、または識別子でログをフィルタリングできます。詳細は、Web コンソールでのログのフィルタリング を参照してください。

6.4.2. Web コンソールでのログのフィルタリング

Web コンソールでログエントリーをフィルタリングできます。

前提条件

- RHEL 9 Web コンソールがインストールされている。
- cockpit サービスが有効になっている。
- ユーザーアカウントが Web コンソールにログインできる。
  手順は、Web コンソールのインストールおよび有効化 を参照してください。

手順

1. RHEL 9 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。
3. コンソールデフォルトでは、Web コンソールには最新のログエントリーが表示されます。別の時間範囲でフィルタリングするには、**Time** ドロップダウンメニューをクリックして、希望するオプションを選択します。
4. 重大度ログのリストは、デフォルトで **エラー以上のレベル** が表示されます。優先度のフィルタリングを変更するには、ドロップダウンメニューの **エラー以上のレベル** をクリックして、優先度を選択します。
5. デフォルトでは、Web コンソールにはすべての識別子のログが表示されます。特定のサービスのログをフィルタリングするには、**All** ドロップダウンメニューをクリックして、識別子を選択します。
6. ログエントリーを開くには、選択したログをクリックします。

6.4.3. Web コンソールでログをフィルターするためのテキスト検索オプション

テキスト検索オプション機能では、ログをフィルタリングするためのさまざまなオプションを利用できます。テキスト検索を使用してログをフィルタリングする場合は、3 つのドロップダウンメニューに定義した事前定義オプションを使用するか、自分で検索全体を入力できます。

ドロップダウンメニュー

検索のメインパラメーターを指定するのに使用できるドロップダウンメニューには、以下の 3 つがあります。

- **時間**: このドロップダウンメニューには、検索のさまざまな時間範囲が事前定義されます。

- **優先度**: このドロップダウンメニューでは、さまざまな優先度のオプションを利用できます。**journalctl --priority** オプションに対応します。デフォルトの優先度の値は Error and above です。これは、他の優先度を指定しないと毎回設定されます。
- **識別子**: このドロップダウンメニューでは、フィルタリングする ID を選択します。**journalctl --identifier** オプションに対応します。

量記号

検索を指定するのに使用できる量記号は 6 つあります。これらは、ログテーブルをフィルタリングするための Options で説明されています。

ログフィールド

特定のログフィールドを検索する場合は、フィールドとその内容を指定することができます。

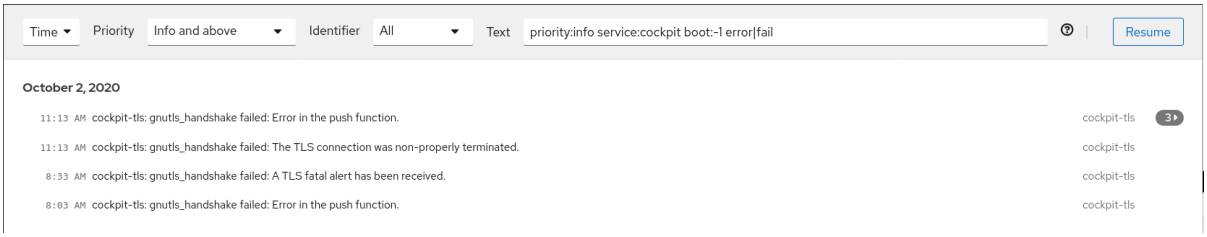
ログメッセージでの自由形式のテキスト検索

ログメッセージで任意のテキスト文字列をフィルタリングできます。文字列は、正規表現の形式にすることもできます。

高度なログのフィルタリング I

2020 年 10 月 22 日深夜以降に発生した 'systemd' によって識別されるすべてのログメッセージをフィルターします。ジャーナルフィールド 'JOB_TYPE' は 'start' または 'restart' のいずれかです。

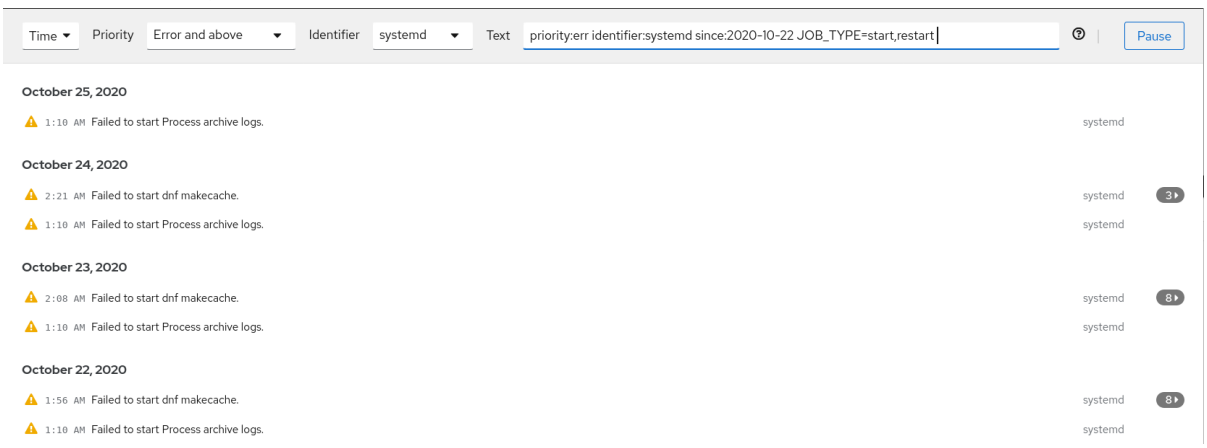
1. フィールドを検索するには、**identifier:systemd since:2020-10-22 JOB_TYPE=start,restart** と入力します。
2. 結果を確認します。



高度なログフィルタリング II

最後の前に起動で 'cockpit.service' systemd ユニットから送信されたすべてのログメッセージ、およびメッセージのボディーに "error" または "fail" が含まれるログメッセージをすべてフィルターします。

1. 検索フィールドに **service:cockpit boot:-1 error|fail** と入力します。
2. 結果を確認します。



6.4.4. テキストボックスのボックスを使用した Web コンソールでのログのフィルター

Web コンソールのテキスト検索ボックスを使用して、さまざまなパラメーターに基づいてログをフィルタリングできます。検索では、フィルタリングドロップダウンメニュー、数量詞、ログフィールド、および自由形式の文字列検索を組み合わせて使用します。

前提条件

- RHEL 9 Web コンソールがインストールされている。
- cockpit サービスが有効になっている。
- ユーザーアカウントが Web コンソールにログインできる。
  手順は、Web コンソールのインストールおよび有効化 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。
   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。
3. ドロップダウンメニューを使用して、3 つの主要なフィルタリング対象の数量 (時間範囲、優先順位、識別子) (フィルタリングする) を指定します。
   優先順位 の数量には常に値が必要です。これを指定しない場合、**Error and above** の優先度が自動的にフィルターされます。テキスト検索ボックスに、設定したオプションに注目してください。
4. フィルターするログフィールドを指定します。
   ログフィールドは複数追加できます。
5. 自由形式文字列を使用して他の文字を検索できます。検索ボックスにも正規表現も使用できます。

6.4.5. ログフィルタリングのオプション

複数の **journalctl** オプションがあり、Web コンソールでのログのフィルタリングに使用できます。これは便利な場合があります。これらの一部は、Web コンソールインターフェイスのドロップダウンメニューですでに扱われています。

表6.4 表

オプション名	用途	備考
priority	メッセージの優先度による出力をフィルタリングします。単一数値またはテキストログレベルを取ります。ログレベルは、通常の syslog ログレベルです。単一のログレベルが指定されている場合、このログレベルまたは低い (より重要な) ログレベルを持つすべてのメッセージが表示されます。	**優先順位** ドロップダウンメニューで説明されます。

オプション名	用途	備考
identifier	指定された syslog 識別子 SYSLOG_IDENTIFIER のメッセージを表示します。複数回指定できます。	**識別子** ドロップダウンメニューで説明されています。
follow	最新のジャーナルエントリーのみを表示し、ジャーナルに追加されるように新しいエントリーを継続的に出力します。	ドロップダウンで説明しません。
service	指定した **systemd** ユニットのメッセージを表示します。複数回指定できます。	ドロップダウンで説明されません。**journalctl --unit** パラメーターに対応します。
boot	特定のブートのメッセージを表示します。 正の整数は、ジャーナルの最初から起動を探し、ゼロ以下の整数は、ジャーナルの最後から起動を探します。このため、1 は、時系列順でジャーナルで見つかった最初の起動を意味し、2 は次に見つかったものと続きます。また、-0 は最後の起動、-1 は最後の起動の 1 つ前などとなります。	**時間** ドロップダウンメニューでは、**現在の起動** または **以前の起動** としてのみ説明されています。その他のオプションは手動で書き込む必要があります。
since	指定の日付以降のエントリーまたは指定の日付以前のエントリーを示します。日付は、"2012-10-30 18:17:16" の形式にする必要があります。時間部分を省略すると、"00:00:00" が想定されます。2 番目のコンポーネントのみを省略すると、":00" が想定されます。日付コンポーネントを省略すると、現在の日付が想定されます。"yesterday"、"today"、"tomorrow" も利用できます。それぞれ、現在の日付けの前の 00:00:00、現在の日付け、現在の日付けの後の日を参照します。"now" は、現在時刻を意味します。最後に、相対時間は"-" または "+" を前に付けてを指定できます。これは、現在時間の前または後の時間を参照します。	ドロップダウンで説明しません。

6.5. 関連情報

- システム上の **journalctl(1)** man ページ
- リモートロギングソリューションの設定

第7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、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。

7.1. 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。

7.1.1. 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。

7.1.2. 予約ユーザーおよびグループ ID の設定

デフォルトでは、RHEL は 1000 未満のユーザー 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ます。予約済みのユーザー ID とグループ ID は、**setup** パッケージで確認できます。**UID_MIN** および **GID_MIN** 値を変更する前に作成されたユーザーおよびグループの UID および GID は変更されません。予約済みのユーザー ID とグループ ID については、次のドキュメントに記載されています。

```
/usr/share/doc/setup/uidgid
```

将来的に予約済みの範囲が拡大する可能性があるため、新しいユーザーとグループには、5000 以降の ID を割り当てます。

デフォルト (1000) 以外の開始 ID を定義するには、**/etc/login.defs** ファイルの **UID_MIN** および **GID_MIN** パラメーターを変更します。

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。

手順

1. エディターで **/etc/login.defs** ファイルを開きます。
2. **UID_MIN** 変数を設定します。次に例を示します。

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

3. **GID_MIN** 変数を設定します。次に例を示します。

```
# 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。

7.1.3. ユーザープライベートグループ

RHEL では **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、Linux グループの管理が容易です。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。

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

7.2. 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。ユーザーとグループ、システムユーザーとシステムグループの最小/最大 ID を定義するには、**/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。

7.2.1. 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。

手順

- ユーザーアカウントを新規作成します。

```
# 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

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

7.3. 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。

7.3.1. 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** アクセス権がある。

手順

- 新しいユーザーを追加するには、次のコマンドを使用します。

```
# 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7.1 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

- 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
```

このコマンドは以下の出力を返します。

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

関連情報

- **useradd** の man ページ

7.3.2. 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** アクセス権がある。

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7.2 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

- 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。

```
# getent group sysadmin
```

このコマンドは以下の出力を返します。

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

関連情報

- **groupadd** の man ページ

7.3.3. 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** アクセス権がある。

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>
```

検証

- 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 <username>
```

7.3.4. 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。

前提条件

- **Root** アクセス権がある。

手順

1. ディレクトリーを作成します。

```
# mkdir <directory-name>
```

2. グループを作成します。

```
# groupadd <group-name>
```

3. ユーザーをグループに追加します。

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

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>
```

5. ユーザーがファイルやディレクトリーを作成および変更できるように書き込み権限を設定し、この権限がディレクトリー内に適用されるように **setgid** ビットを設定します。

```
# chmod g+rwxs <directory>
```

検証

- 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>
```

このコマンドは以下の出力を返します。

```
*drwx__rws__r-x.* 2 root _group-name_ 6 Nov 25 08:45 _directory-name_
```

7.3.5. 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。また、ユーザーアカウントを削除し、必要に応じてホームディレクトリーや設定ファイルなどのユーザーデータとメタデータを削除するコマンドを以下に示します。

- **root** アクセス権がある。
- 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
```

- 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。

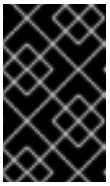
  a. ユーザー、そのホームディレクトリー、メールスプール、および SELinux ユーザーマッピングを削除します。

```
# 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

7.4. 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、システムユーザーアカウントを追加、編集、削除するためのグラフィカルインターフェイスを備えています。

Web コンソールでパスワードの有効期限を設定したり、ユーザーセッションを終了したりすることもできます。

7.4.1. Web コンソールを使用した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。

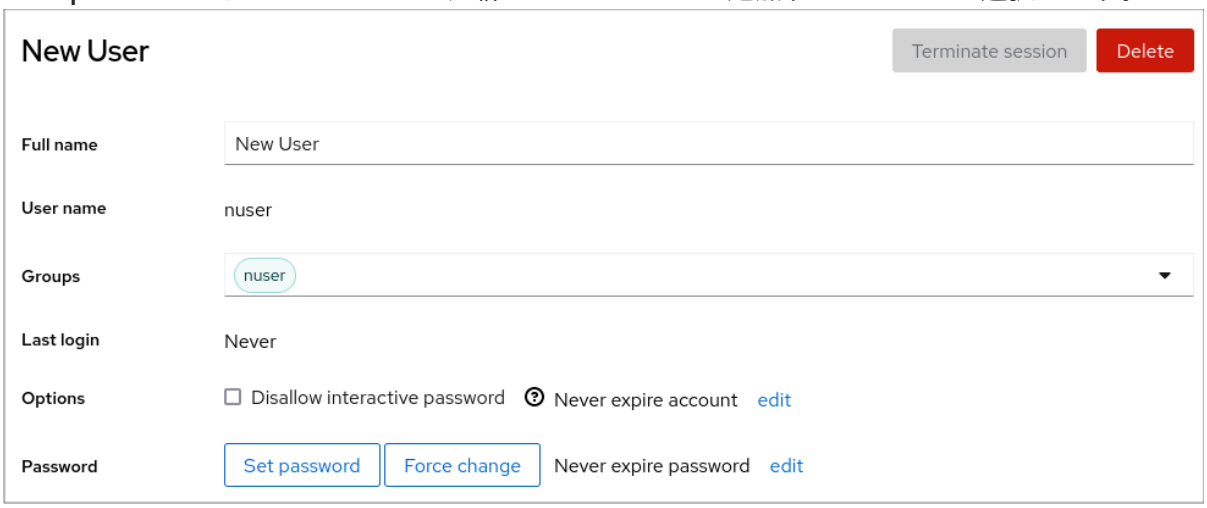
前提条件

- RHEL 9 Web コンソールがインストールされている。
- cockpit サービスが有効になっている。
- ユーザーアカウントが Web コンソールにログインできる。
  手順は、Web コンソールのインストールおよび有効化 を参照してください。

手順

1. RHEL 9 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. **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。



検証

- 新しいアカウントを **Accounts** 設定で確認できます。その認証情報を使用してシステムに接続できます。

7.4.2. Web コンソールでパスワード有効期限の強制

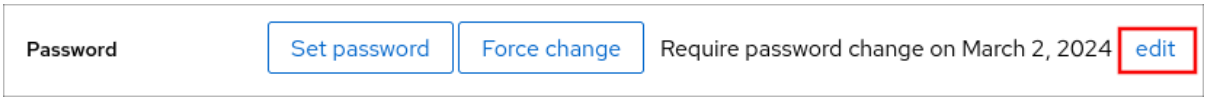
デフォルトでは、ユーザーアカウントのパスワードに期限はありません。定義した日数が経過したら、システムパスワードが期限切れになるように設定できます。パスワードの有効期限が切れると、ユーザーは次回のログイン時に、システムにアクセスする前にパスワードを変更する必要があります。

前提条件

- RHEL 9 Web コンソールがインストールされている。
- cockpit サービスが有効になっている。
- ユーザーアカウントが Web コンソールにログインできる。
  手順は、Web コンソールのインストールおよび有効化 を参照してください。

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までの日数 (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。

7.5. 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。

7.5.1. 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。

RHEL では、ユーザーグループはプライマリーまたは補助グループ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、0 個、1 個、または複数の補助グループのメンバーになることができます。

7.5.2. 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
```

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7.3 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。

7.5.3. 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>
```

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。

- 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 <username>
```

7.5.4. 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** アクセス権がある。

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>
```

検証

- 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 <username>
```

7.5.5. 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** アクセス権がある。

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>
```

検証

- 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 <username>
```

7.5.6. 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** アクセス権がある。
- 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>
```

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。

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。

検証

- 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 <username>
```

7.6. 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くなった場合は、**root** ユーザーでも root 以外のユーザーでも、root パスワードを変更できます。

7.6.1. 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** アクセス権がある。

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。

7.6.2. 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。
- **sudo** を使用して root としてコマンドを実行する権限がある。

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。

7.6.3. root パスワードのリセット

root ユーザーとしてログインできず、sudo 権限を持つ root 以外ユーザーがない場合は、root パスワードをリセットできます。管理 **wheel** グループに属していない場合は、システムを特殊モードで起動して root パスワードをリセットできます。このモードでは、システムが **initramfs** から実際のシステムに制御を渡す前に、ブートプロセスが停止します。

手順

1. システムを再起動し、GRUB ブート画面で **e** キーを押してブート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. カーソルを **linux** で始まる行の末尾にセットします。
3. **linux** で始まる行の末尾に **rd.break** を追加します。
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
   ```

   デフォルトでは、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
   ```
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. システムを再起動して、**switch_root** プロンプトを終了します。

    ```
    exit
    ```
11. SELinux の再ラベル付けプロセスが完了するまで待ちます。大きなディスクの再ラベル付けには長い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

1. 新しい root パスワードを使用して、**root** ユーザーとしてログインします。

2. オプション: 現在の有効なユーザー ID に関連付けられているユーザー名を表示します。

```
# whoami
```

第8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。

8.1. SUDOERS のユーザー認可

/etc/sudoers ファイルと、デフォルトで **/etc/sudoers.d/** ディレクトリー内にあるドロップインファイルでは、**sudo** コマンドを使用して他のユーザーとしてコマンドを実行できるユーザーを指定します。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。

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>
```

詳細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

ルールの一部または複数のセグメントで ALL を使用すると、重大なセキュリティーリスクが発生する可能性があります。

! 演算子を使用して引数を否定することができます。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

- **sudoers(5)** man ページ

8.2. グループのメンバーが ROOT としてコマンドを実行できるようにするための SUDO ルールを追加する

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。

- sudoers 設定ファイルにリスト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。ただし、例外があります。たとえば、すべてのユーザーに完全な **sudo** 権限が付与されている場合などです。このような場合、ユーザーは root シェルでコマンドを切り替えて実行できます。以下に例を示します。
- **sudo -i**
- **sudo su -**

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

- **wheel** グループのメンバーとしてログインし、次のコマンドを実行します。

```
# sudo whoami
root
```

関連情報

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

8.3. 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。これを行うと、ユーザーに完全な **sudo** アクセス権を付与したり、ユーザーに root パスワードを付与したりするよりも、セキュリティーが向上します。その理由は次のとおりです。

- 特権を必要とする操作のより細かい制御。ユーザーに完全な管理アクセス権を付与せずに、特定のホストで特定の操作を実行することを許可できます。
- ロギングの改善。ユーザーが **sudo** を通じて操作を実行したときに、その操作が root だけでなくユーザー名とともにログに記録されます。
- 透明性のある制御。ユーザーが **sudo** 権限の使用を試みるたびにメール通知を送信するように設定できます。

前提条件

- システムへの root アクセス権があります。

手順

1. **/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>
```

ファイルがエディターで自動的に開きます。

2. 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、コマンドの実行許可に関連す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 **dnf** コマンドと **reboot** コマンドを **host1.example.com** で実行できるようにするには、次のように入力します。

```
user1 host1.example.com = /bin/dnf, /sbin/reboot
```

1. オプション: ユーザーが **sudo** 権限を使用しようとするたびにメール通知を受け取るには、ファイルに次の行を追加します。

-

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

2. 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 whoami
[sudo] password for <username>:
```

ユーザーの **sudo** パスワードを入力します。

3. 権限が正しく設定されている場合、sudo は設定されたユーザーとしてコマンドを実行します。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
...
```

システムから次のエラーメッセージが返された場合、sudo を使用してコマンドを実行することがユーザーに許可されていません。

```
<username> is not in the sudoers file. This incident will be reported.
```

+ システムから次のエラーメッセージが返された場合、設定が正しく完了していません。

```
<username> is not allowed to run sudo on <host.example.com>.
```

+ システムから次のエラーメッセージが返された場合、ユーザーのルールでコマンドが正しく定義されていません。

```
`Sorry, user _<username>_ is not allowed to execute '_<path/to/command>_' as root on
_<host.example.com>_.`
```

関連情報

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

8.4. RHEL システムロールを使用したカスタム SUDOERS 設定の適用

sudo RHEL システムロールを使用して、管理対象ノードにカスタムの **sudoers** 設定を適用できます。これにより、どのユーザーがどのホストでどのコマンドを実行できるかを定義することで、設定の効率が向上し、よりきめ細かい制御が可能になります。

前提条件

- コントロールノードと管理対象ノードの準備が完了している。
- 管理対象ノードで Playbook を実行できるユーザーとしてコントロールノードにログインしている。
- 管理対象ノードへの接続に使用するアカウントに、そのノードに対する **sudo** 権限がある。

手順

1. 次の内容を含む Playbook ファイル (例: ~/**playbook.yml**) を作成します。

```
---
- name: "Configure sudo"
  hosts: managed-node-01.example.com
  tasks:
    - name: "Apply custom /etc/sudoers configuration"
      ansible.builtin.include_role:
        name: redhat.rhel_system_roles.sudo
      vars:
        sudo_sudoers_files:
          - path: "/etc/sudoers"
            user_specifications:
              - users:
                  - <user_name>
                hosts:
                  - <host_name>
                commands:
                  - <path_to_command_binary>
```

Playbook で指定する設定は次のとおりです。

users

ルールを適用するユーザーのリスト。

hosts

ルールを適用するホストのリスト。すべてのホストの場合は **ALL** を使用できます。

commands

ルールを適用するコマンドのリスト。すべてのコマンドの場合は **ALL** を使用できます。

Playbook で使用されるすべての変数の詳細は、コントロールノードの **/usr/share/ansible/roles/rhel-system-roles.sudo/README.md** ファイルを参照してください。

2. Playbook の構文を検証します。

```
$ ansible-playbook --syntax-check ~/playbook.yml
```

このコマンドは構文を検証するだけであり、有効だが不適切な設定から保護するものではないことに注意してください。

3. Playbook を実行します。

```
$ ansible-playbook ~/playbook.yml
```

検証

1. 管理対象ノードで、Playbook によって新しいルールが適用されたことを確認します。

```
# cat /etc/sudoers | tail -n1
<user_name> <host_name>= <path_to_command_binary>
```

関連情報

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

第9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。

9.1. 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。
- **ユーザーファイル作成モードマスク** (umask)

9.1.1. 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9.1 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9.2 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。
ユーザー所有者以外のユーザーまたはグループが、ディレクトリー内の項目をリスト表示したり、その項目に関する情報にアクセスしたり、変更したりすることはできません。
- **.** 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。

9.1.2. ユーザー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。

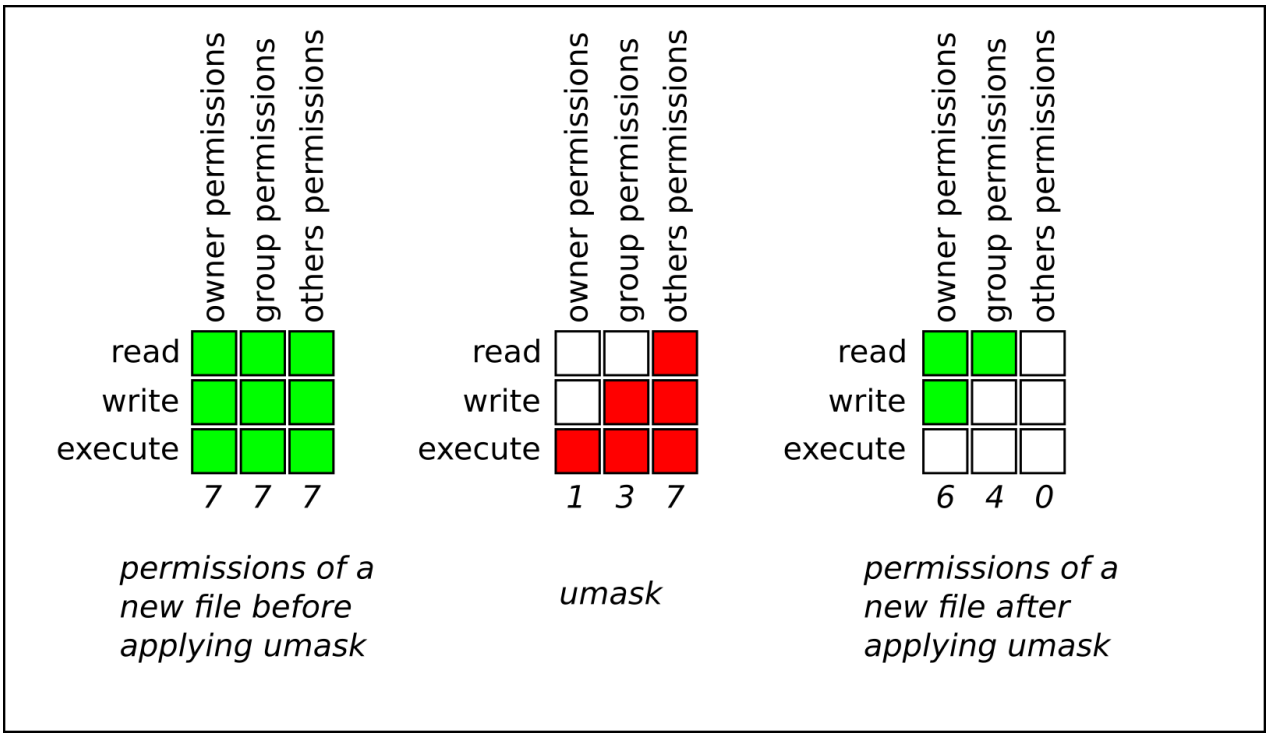
パーミッション	シンボリック値	8 進数値
読み取り、書き込み、実行	rwx	0
読み取りおよび書き込み	rw-	1
読み取りおよび実行	r-x	2
読み取り	r--	3
書き込みおよび実行	-wx	4
書き込み	-w-	5
実行	--x	6
権限なし	---	7

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9.3 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。



9.1.3. 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9.4 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9.5 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は **644** (**-rw-r—r--**) になりました。

	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。

9.1.4. シンボリック値を使用したファイル権限の変更

chmod ユーティリティーをシンボリック値 (文字と記号の組み合わせ) とともに使用して、ファイルまたはディレクトリーのファイル権限を変更できます。

次の **権限** 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

- 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9.6 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
```

9.1.5. 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。

9.2. 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。

9.2.1. 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** アクセス権がある。

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、次のコマンドを実行します。

```
$ 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。

- 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、システム上の **setfacl** man ページを参照してください。

例9.7 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

- 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--
```

9.3. 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。

9.3.1. 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。

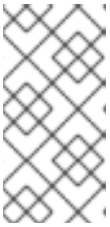
手順

- 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。

9.3.2. 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。

次の **権限** を割り当てることができます。

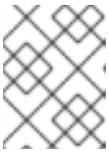
- 読み取り (r)
- 書き込み (w)
- 実行 (x)

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。

9.3.3. 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。

9.3.4. 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** アクセス権がある。

手順

1. エディターで **/etc/bashrc** ファイルを開きます。

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。

1. 変更を保存し、エディターを終了します。

9.3.5. 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。

9.3.6. 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。

9.3.7. 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10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。

10.1. 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10.1 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。
/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
```

10.2. 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。

10.2.1. 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

10.2.2. 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

- 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10.2 利用可能なサービスユニットの情報

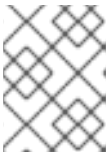
フィールド	説明
Loaded	サービスユニットがロードされているかどうかの説明、ユニットファイルへの絶対パス、およびブート時のユニット起動が有効かどうかの注記。
Active	サービスユニットが実行中かどうかの説明と、タイムスタンプ
Main PID	プロセス ID と、対応するシステムサービスの名前。
ステータス	対応するシステムサービスに関する追加情報
Process	関連プロセスに関する追加情報
CGroup	関連するコントロールグループ (**cgroups**) に関する追加情報。

- 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

10.2.3. systemd ユニットの起動と停止

systemctl 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 アクセス権がある。

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 <systemd_unit> *
```

<systemd_unit> は、起動するサービスユニットの名前 (例: **httpd.service**) 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、**sendmail** サービスを実行しているときに **postfix** サービスを起動しようとすると、**systemd** はまず **sendmail** を自動的に停止します。これら 2 つのサービスは、競合しており、同じポートで実行できないためです。

関連情報

- システム上の **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

10.2.4. システムサービスの停止

現在のセッションでシステムサービスを停止する場合は、**systemctl 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

- システム上の **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

10.2.5. システムサービスの再起動と再ロード

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 アクセス権がある。

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。

選択したサービスユニットが実行中でない場合は、このコマンドによってサービスユニットが起動されます。

- 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
```

- 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

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

10.2.6. 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 アクセス権がある。

手順

- ユニットがマスクされているかどうかを確認します。

```
# systemctl status <systemd_unit>
```

- ユニットがマスクされている場合は、まずマスクを解除します。

```
# systemctl unmask <systemd_unit>
```

- システムの起動時に起動するようにサービスを有効にします。

```
# systemctl enable <systemd_unit>
```

<systemd_unit> は、有効にするサービスユニットの名前 (例: **httpd**) に置き換えます。

必要に応じて、コマンドに **--now** オプションを渡すと、ユニットがすぐに起動します。

関連情報

- システム上の **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

10.2.7. 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 アクセス権がある。

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。必要に応じて、**--now** コマンドを渡すと、サービスが現在実行中であれば停止させることができます。

- オプション: ユニットが管理者によって誤って起動されたり、他のユニットの依存関係として起動されたりするのを防ぐために、サービスをマスクします。

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

関連情報

- システム上の **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

10.3. 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。

10.3.1. 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。

以下の例を考慮してください。

- 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10.3 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

- システム上の **systemd.special(7)** および **systemd.target(5)** man ページ

10.3.2. ブート先のデフォルトターゲットの変更

default.target シンボリックリンクは、システムのブート先の systemd ターゲットを参照します。システムが起動すると、systemd はこのリンクを解決し、定義されたターゲットで起動します。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 アクセス権がある。

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. オプション: 新しいデフォルトターゲットに切り替えます。

```
# systemctl isolate default.target
```

または、システムを再起動します。

関連情報

- システム上の **systemctl(1)**、**systemd.special(7)**、**bootup(7)** man ページ

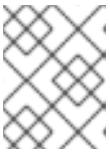
10.3.3. 現在のターゲットの変更

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。手動で別のターゲットに切り替えるのは一時的な操作にすぎません。ホストをリブートすると、systemd はデフォルトのターゲットで再度起動します。

手順

1. 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。

2. 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

- システム上の **systemctl(1)** man ページ

10.3.4. 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中の全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

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。

10.3.5. 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. 別のブート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。

10.4. 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。

10.4.1. 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 ユーティリティーを使用すると、次の利点があります。

- RHEL 8 では、**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。

10.4.2. 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する

  ```
  # 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

10.4.3. 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

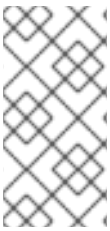
systemctl コマンドを使用して、次のタスクのいずれかを実行します。

- 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。

10.4.4. 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。

前提条件

- Root アクセス権がある。

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。

10.4.5. 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
- ハイバネート
- ハイブリッドスリープ
- サスペンドしてからハイバネート

前提条件

- Root アクセス権がある。

手順

適切な省電力方法を選択します。

- **Suspend**: サスペンドを実行すると、システムの状態が RAM に保存され、RAM モジュールを除いて、マシン内のほとんどのデバイスの電源がオフになります。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。システムをサスペンドするには、次のコマンドを実行します。

```
# systemctl suspend
```

- **Hibernate**: ハイバネートを実行すると、システムの状態がハードディスクドライブに保存され、マシンの電源がオフになります。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。システムをハイバネートするには、次のコマンドを実行します。

```
# systemctl hibernate
```

- **Hybrid sleep**: 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。システムをハイバネートおよびサスペンドするには、次のコマンドを実行します。

```
# systemctl hybrid-sleep
```

- **Suspend-then-hibernate**: 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。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
```

10.4.6. 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。

10.4.6.1. GNOME が起動していないときに電源ボタンを押したときの動作の変更

非グラフィカルな **systemd** ターゲットの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** 設定ファイルを編集し、**HandlePowerKey=poweroff** 変数を次のいずれかのオプションに設定します。

   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

   何もしません。

   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
   ```

10.4.6.2. GNOME が起動しているときに電源ボタンを押したときの動作の変更

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、デフォルトではマシンがサスペンドします。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。

手順

1. 次の内容で、**/etc/dconf/db/local.d/01-power** ファイルにシステム全体の設定用のローカルデータベースを作成します。

   ```
   [org/gnome/settings-daemon/plugins/power]
   power-button-action=<value>
   ```

   <value> を次のいずれかの電源ボタンアクションに置き換えます。

   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。
interactive モードでは、電源ボタンを押すと 60 秒後にシステムの電源が自動的にオフになります。ただし、ポップアップクエリーとは異なる動作を選択することもできます。

2. オプション: ユーザーの設定をオーバーライドし、ユーザーが変更できないようにします。**/etc/dconf/db/local.d/locks/01-power** ファイルに次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ために、ログアウトして再度ログインします。

第11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。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。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。

11.1. CHRONY スイートの概要

Network Time Protocol (NTP) の実装は chrony 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使用

chrony はさまざまな環境で優れたパフォーマンスを発揮します。

- 断続的なネットワーク接続
- 非常に混雑したネットワーク
- 温度の変化 (通常のコンピューターの時計は温度の影響を受けます)
- 継続的に実行されないシステム、または仮想マシン上で実行されるシステム

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。

chronyd デーモンは、コマンドラインユーティリティー **chronyc** によって監視および制御できます。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。

11.2. 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。

11.3. CHRONY の使用

次のセクションでは、**chronyd** を起動および停止する方法と、**chrony** が同期されているかどうかを確認する方法について説明します。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。

11.3.1. chrony の管理

chronyd を起動、停止し、そのステータスを確認できます。

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
```

11.3.2. chrony の同期確認

chrony が同期されているかどうかは、**tracking**、**sources**、**sourcestats** コマンドを使用して確認できます。

手順

1. **chrony** のトラッキングを確認するには、次のように入力します。

```
$ 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. **chronyc** の sources コマンドは、**chronyd** がアクセスしている現在の時間ソースに関する情報を表示します。

```
$ 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

- システム上の **chronyc(1)** man ページ

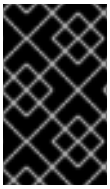
11.3.3. システムクロックの手動調整

システムクロックは手動で調整できます。

手順

- システムクロックを徐々に調整していく (slew) のをやめ、一度に修正 (step) するには、次のように入力します。

```
# chronyc makestep
```



重要

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。

11.3.4. 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。

NetworkManager は、インターフェイスの再設定、停止、または起動操作中に **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。

手順

- **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 -f -s /dev/null /etc/NetworkManager/dispatcher.d/20-chrony-onoffline
```

注記

この変更後、NTP サーバーは常にオンライン状態になります。

11.3.5. 分離されたネットワークでの 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。

分離されたネットワーク内のシステムに **chrony** を設定するには、次の手順に従います。

手順

1. サーバーとして選択したシステムで、**/etc/chrony.conf** を次のように編集します。

```
driftfile /var/lib/chrony/drift
commandkey 1
keyfile /etc/chrony.keys
initstepslew 10 client1 client3 client6
local stratum 8
manual
allow <subnet>
```

<subnet> は、クライアントの接続元として許可するネットワークです。サブネットを指定するには、Classless Inter-Domain Routing (CIDR) 表記を使用します。

2. サーバーの直接のクライアントとして選択したシステムで、**/etc/chrony.conf** を次のように編集します。

```
server <server_fqdn>
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 <server_ip_address>
```

<server_fqdn> はサーバーのホスト名、**<server_ip_address>** はサーバーのアドレスです。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。

11.3.6. リモート監視アクセスの設定

chronyc ユーティリティーは、次の方法を使用して **chronyd** にアクセスできます。

- IPv4 または IPv6
- **root** および **chrony** ユーザーがローカルでアクセスできるドメインソケット

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。この接続が失敗した場合、**chronyc** は 127.0.0.1 に接続を試み、さらに ::1 に接続を試みます。

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

これらのコマンドは **chronyd** が受信します。コマンドの受信元とするホストのセットを、次の方法を使用して設定できます。

- **chronyd** の設定ファイルで **cmdallow** ディレクティブを使用できます。
- **chronyc** で **cmdallow** コマンドを実行します。

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. **/etc/chrony.conf** ファイルに以下を追加して、**chrony** がローカルインターフェイスでリッスンするように設定します。

```
bindcmdaddress 0.0.0.0
```

および

```
bindcmdaddress ::
```

2. リモート IP アドレス、ネットワーク、サブネットからのコマンドを許可します。**/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
```

```
cmdallow 2001:db8::/64
```

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の接続を許可します。

```
# firewall-cmd --permanent --add-port=323/udp
```

4. 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

- システム上の **chrony.conf(5)** man ページ

11.3.7. 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11.1 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

11.3.8. 関連情報

- システム上の **chronyc(1)** および **chronyd(8)** man ページ
- よくある質問 (FAQ)

11.4. ハードウェアのタイムスタンプを使用した CHRONY

一部のネットワークインターフェイスコントローラー (NIC) のハードウェアタイムスタンプ (HW) は、着信パケットと発信パケットの正確なタイムスタンプを提供します。**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。最高の同期精度を実現する場合は、**PTP** をサポートするスイッチやルーターがあるネットワークでは **PTP** を使用し、そのようなスイッチやルーターがないネットワークでは **NTP** を選択してください。

11.4.1. 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。

手順

- デバイスのタイムスタンプ機能と、関連する PTP ハードウェアクロックを表示します。

```
# ethtool -T enp1s0
```

11.4.2. ハードウェアのタイムスタンプの有効化

/etc/chrony.conf ファイルの **hwtimestamp** ディレクティブを使用して、1つまたは複数のインターフェイスでハードウェアタイムスタンプを有効にできます。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。

手順

1. **/etc/chrony.conf** ファイルを編集し、次の変更を加えます。

   a. ハードウェアタイムスタンプをサポートするインターフェイスに **hwtimestamp** 設定を追加します。以下に例を示します。

   ```
   hwtimestamp enp1s0
   hwtimestamp eno*
   ```

   ptp4l などの他のアプリケーションがハードウェア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 * を使用できます。

   b. **minpoll** および **maxpoll** オプションをサーバー設定に追加して、短いクライアントポーリング間隔を設定します。次に例を示します。

   ```
   server ntp.example.comlocal minpoll 0 maxpoll 0
   ```

   ハードウェアタイムスタンプの場合、システムクロックのオフセットを最小限に抑えるために、デフォルトの範囲 (64 - 1024 秒) よりも短いポーリング間隔を設定する必要があります。

   c. サーバー設定に **xleave** オプションを追加して、NTP インターリーブモードを有効にします。

   ```
   server ntp.example.comlocal minpoll 0 maxpoll 0 xleave
   ```

   この設定では、chrony はパケットを送信した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、サーバーが応答するパケット内のタイムスタンプを、サーバーが保存できなくなります。**xleave** オプションを使用すると、chrony は送信後に生成された送信タイムスタンプを受信できます。

   d. オプション: サーバーでクライアントのアクセスのロギング用に割り当てられるメモリーの最大サイズを増やします。次に例を示します。

   ```
   clientloglimit 100000000
   ```

   デフォルトのサーバー設定では、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。**clientloglimit** 設定の値を増やすことで、多数のクライアントに対応するサーバーを設定できます。

2. chronyd サービスを再起動します。

   ```
   # systemctrl restart chronyd
   ```

検証

1. オプション: **/var/log/messages** ログファイルでハードウェアタイムサンプリングが有効になっていることを確認します。

```
chronyd[4081]: Enabled HW timestamping on enp1s0
chronyd[4081]: Enabled HW timestamping on eno1
```

2. 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、送信および受信タイムスタンプモードとインターリーブモードを表示します。

```
# chronyc ntpdata

Output:

[literal,subs="+quotes,verbatim,normal"]


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
```

3. NTP 測定の安定性を表示します。

```
# chronyc sourcestats

Output:

[literal,subs="+quotes,verbatim,normal"]
....

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
....
```

この安定性は、**Std Dev** 列に表示されます。ハードウェアタイムスタンプを有効にすると、NTP 測定の安定性は、通常の負荷で数十または数百ナノ秒になります。

11.4.3. 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。

手順

1. 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。

1. **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。

```
bindaddress 203.0.113.74
hwtimestamp enp1s0
local stratum 1
```

2. chronyd サービスを再起動します。

```
# systemctrl restart chronyd
```

11.5. 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。**/etc/sysconfig/chronyd** ファイルに **GNUTLS_FORCE_FIPS_MODE=0** 設定を追加することで、**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。

11.5.1. クライアントでの 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

- タイムサーバーが NTS をサポートしている。

手順

/etc/crony.conf ファイルを編集し、次の変更を加えます。

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. システムの起動中に Network Time Security-Key Establishment (NTS-KE) セッションが繰り返されないように、次の設定を追加します。

```
ntsdumpdir /var/lib/chrony
```

3. 次の設定をコメントアウトまたは削除して、**DHCP** によって提供される Network Time Protocol (NTP) サーバーとの同期を無効にします (設定が存在する場合)。

```
sourcedir /run/chrony-dhcp
```

4. **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.netnod.se   NTS     1   15  256  33m    0    0    8  100
ptbtime1.ptb.de   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.netnod.se   1       6   377    44   +237us[ +237us] +/-   23ms
ptbtime1.ptb.de   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

- システム上の **chrony.conf(5)** man ページ

11.5.2. タイムサーバーでの Network Time Security (NTS) 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. **/etc/chrony.conf** ファイルを編集し、次の変更を加えます。

```
ntsserverkey /etc/pki/tls/private/<ntp-server.example.net>.key
ntsservercert /etc/pki/tls/certs/<ntp-server.example.net>.crt
```

2. 秘密鍵と証明書ファイルの両方に、ファイルの読み取りを chrony ユーザーに許可する権限を設定します。次に例を示します。

```
# chown root:chrony /etc/pki/tls/private/<ntp-server.example.net>.key
/etc/pki/tls/certs/<ntp-server.example.net>.crt

# chmod 644 /etc/pki/tls/private/<ntp-server.example.net>.key /etc/pki/tls/certs/<ntp-server.example.net>.crt
```

3. **ntsdumpdir /var/lib/chrony** 設定が存在することを確認します。

4. Firewalld で必要なポートを開きます。

```
# firewall-cmd --permannent --add-port={323/udp,4460/tcp}
# firewall-cmd --reload
```

5. **chronyd** を再起動します。

```
# systemctl restart chronyd
```

検証

1. クライアントマシンから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。

2. 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12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。

12.1. REAR の設定とバックアップの手動作成

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. 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 ReaR 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。

12.2. 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

RHEL 9.2 以降、基本的な Relax and Recover (ReaR) 機能が 64 ビットの IBM Z アーキテクチャーで利用できるようになり、完全にサポートされ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

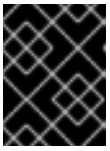
以下の変数を**/etc/rear/local.conf**に追加し、64 ビット IBM Z アーキテクチャーでレスキューイメージを生成するように ReaR を設定します。

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. オプション: **OUTPUT_URL** 変数を設定して、カーネルファイルと **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

12.3. REAR の除外

ReaR ユーティリティーは、復旧プロセス中に、**/var/lib/rear/layout/disklayout.conf** レイアウトファイルの内容に従って、レスキューイメージが作成された元のシステムのストレージレイアウトを、復旧されたシステムのディスク上に再作成します。ストレージレイアウトには、パーティション、ボリュームグループ、論理ボリューム、ファイルシステム、その他のストレージコンポーネントが含まれます。

ReaR は、レスキューイメージを作成するときにレイアウトファイルを作成し、このファイルをイメージに埋め込みます。レイアウトファイルは、**rear savelayout** コマンドを使用して作成することもできます。これを使用すると、レスキューイメージ全体を作成しなくても、レイアウトファイルをすばやく作成して調べることができます。

レイアウトファイルには、一部の例外を除き、元のシステムのストレージレイアウト全体が記述されます。ReaR は、復旧中に再作成されないように、一部のストレージコンポーネントをレイアウトファイルから除外します。ストレージコンポーネントをレイアウトから除外するかどうかは、次の設定変数によって制御されます。

- **AUTOEXCLUDE_DISKS**
- **AUTOEXCLUDE_MULTIPATH**
- **AUTOEXCLUDE_PATH**
- **EXCLUDE_RECREATE**

/usr/share/rear/conf/default.conf ファイルで設定変数のデフォルト値を確認できます。このデフォルト値は、ローカルの **/etc/rear/local.conf** 設定ファイルで変更できます。

レイアウトファイルの構文と、一部のストレージコンポーネントを除外するために使用できる設定変数の詳細は、ReaR パッケージとともに **/usr/share/doc/rear/relax-and-recover-user-guide.html** としてインストールされる ReaR ユーザーガイドの **Layout configuration** の章を参照してください。

また、内部の **NETFS** および **RSYNC** バックアップ方式でバックアップされるファイルを設定することもできます。デフォルトでは、レイアウトファイルにファイルシステムが含まれている場合、マウントされたすべてのローカル (ディスクベースの) ファイルシステム上のファイルが、**rear mkbackup** コマンドまたは **rear mkbackuponly** コマンドによってバックアップされます。**AUTOEXCLUDE_DISKS**、**AUTOEXCLUDE_MULTIPATH**、**AUTOEXCLUDE_PATH**、**EXCLUDE_RECREATE** などの変数によって制御されるレイアウトファイルから一部のファイルシステムを除外すると、そのファイルシステムの内容もバックアップから除外されます。また、**BACKUP_PROG_EXCLUDE** 設定変数を使用すると、レイアウトファイルからファイルシステムを除外せずに、一部のファイルまたはディレクトリーツリーをバックアップから除外できます。この方法でファイルシステム内のすべてのファイルとディレクトリーを除外しても、復旧時にファイルシステムが再作成されます。しかし、バックアップに復元するデータが含まれていないため、ファイルシステムは空になります。これは、一時データが含まれており保存する必要のないファイルシステム、または ReaR によらない方法を使用してバックアップされるデータに役立ちます。

BACKUP_PROG_EXCLUDE 変数は、tar または rsync に渡される glob(3) スタイルのワイルドカードパターンの配列です。このパターンは引用符で囲む必要があることに注意してください。これは、シェルが設定ファイルを読み取るときにパターンが拡張されるのを防ぐために行います。この変数のデフォルト値は、**/usr/share/rear/conf/default.conf** ファイルで設定されています。デフォルト値には、たとえば **/tmp/*** パターンが含まれています。このパターンは、**/tmp** ディレクトリーの下にあるすべてのファイルとディレクトリーを除外しますが、**/tmp** ディレクトリー自体は除外しません。

他のファイルやディレクトリーを除外する必要がある場合は、デフォルト値を保持するために、変数をオーバーライドするのではなく、変数にパターンをさらに追加します。たとえば、ディレクトリー **/data/temp** の下にあるすべてのファイルとディレクトリーを除外するには、次のように指定します。

```
# BACKUP_PROG_EXCLUDE+=( '/data/temp/*' )
```

rear mkbackup コマンドを使用すると、ログ内のバックアップ除外パターンがリスト表示されます。ログファイルは **/var/log/rear** ディレクトリーにあります。これを使用して、完全なシステム復旧を実行する前に、除外ルールを確認できます。たとえば、ログに次のエントリーが含まれているとします。

```
2025-04-29 10:17:41.312431050 Making backup (using backup method NETFS)
2025-04-29 10:17:41.314369109 Backup include list (backup-include.txt contents):
2025-04-29 10:17:41.316197323   /
2025-04-29 10:17:41.318052001 Backup exclude list (backup-exclude.txt contents):
2025-04-29 10:17:41.319857125   /tmp/*
2025-04-29 10:17:41.321644442   /dev/shm/*
2025-04-29 10:17:41.323436363   /var/lib/rear/output/*
```

この場合、**/tmp**、**/dev/shm**、**/var/lib/rear/output** ディレクトリーの下にあるすべてのファイルとディレクトリーを除いて、ルートファイルシステム全体がバックアップの対象になります。