Red Hat Enterprise Linux 9

ファイアウォールおよびパケットフィルターの設定

ガイド

Last Updated: 2022-08-08

Red Hat Enterprise Linux 9 ファイアウォールおよびパケットフィルターの設定

ガイド

Enter your first name here. Enter your surname here.
Enter your organisation's name here. Enter your organisational division here.
Enter your email address here.

法律上の通知

概要

目次

多様性を受け入れるオープンソースの強化

Red Hat では、コード、ドキュメント、Web プロパティーにおける配慮に欠ける用語の置き換えに取り組んでいます。まずは、マスター (master)、スレーブ (slave)、ブラックリスト (blacklist)、ホワイトリスト (whitelist) の 4 つの用語の置き換えから始めます。この取り組みは膨大な作業を要するため、今後の複数のリリースで段階的に用語の置き換えを実施して参ります。詳細は、Red Hat CTO である Chris Wright のメッセージをご覧ください。

RED HAT ドキュメントへのフィードバック (英語のみ)

ご意見ご要望をお聞かせください。ドキュメントの改善点はございませんか。

- 特定の部分についての簡単なコメントをお寄せいただく場合は、以下をご確認ください。
    1. ドキュメントの表示が **Multi-page HTML** 形式になっていて、ドキュメントの右上隅に **Feedback** ボタンがあることを確認してください。
    2. マウスカーソルで、コメントを追加する部分を強調表示します。
    3. そのテキストの下に表示される **Add Feedback** ポップアップをクリックします。
    4. 表示される手順に従ってください。
- Bugzilla を介してフィードバックを送信するには、新しいチケットを作成します。
    1. Bugzilla の Web サイトに移動します。
    2. Component で **Documentation** を選択します。
    3. **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記入してください。
    4. **Submit Bug** をクリックします。

第1章 FIREWALLD の使用および設定

ファイアウォール は、外部からの不要なトラフィックからマシンを保護する方法です。**ファイアウォールルール** セットを定義することで、ホストマシンへの着信ネットワークトラフィックを制御できます。このようなルールは、着信トラフィックを分類して、拒否または許可するために使用されます。

firewalld は、D-Bus インターフェースを使用して、動的にカスタマイズできるホストベースのファイアウォールを提供するファイアウォールサービスデーモンです。ルールが変更するたびに、ファイアウォールデーモンを再起動しなくても、ルールの作成、変更、および削除を動的に可能にします。

firewalld は、ゾーンおよびサービスの概念を使用して、トラフィック管理を簡素化します。ゾーンは、事前定義したルールセットです。ネットワークインターフェースおよびソースをゾーンに割り当てることができます。許可されているトラフィックは、コンピューターが接続するネットワークと、このネットワークが割り当てられているセキュリティーレベルに従います。ファイアウォールサービスは、特定のサービスに着信トラフィックを許可するのに必要なすべての設定を扱う事前定義のルールで、ゾーンに適用されます。

サービスは、ネットワーク接続に 1 つ以上のポートまたはアドレスを使用します。ファイアウォールは、ポートに基づいて接続のフィルターを設定します。サービスに対してネットワークトラフィックを許可するには、そのポートを開く必要があります。**firewalld** は、明示的に開いていないポートのトラフィックをすべてブロックします。trusted などのゾーンでは、デフォルトですべてのトラフィックを許可します。

nftables バックエンドを使用した **firewalld** が、**--direct** オプションを使用して、カスタムの **nftables** ルールを **firewalld** に渡すことに対応していないことに注意してください。

1.1. FIREWALLD の使用

本セクションでは、**firewalld** に関する情報を提供します。

1.1.1. firewalld、nftables、または iptables を使用する場合

以下は、次のユーティリティーのいずれかを使用する必要があるシナリオの概要です。

- **firewalld**:簡単なファイアウォールのユースケースには、**firewalld** ユーティリティーを使用します。このユーティリティーは、使いやすく、このようなシナリオの一般的な使用例に対応しています。
- **nftables**:**nftables** ユーティリティーを使用して、ネットワーク全体など、複雑なパフォーマンスに関する重要なファイアウォールを設定します。
- **iptables**:Red Hat Enterprise Linux の **iptables** ユーティリティーは、**legacy** バックエンドの代わりに **nf_tables** カーネル API を使用します。**nf_tables** API は、**iptables** コマンドを使用するスクリプトが、Red Hat Enterprise Linux で引き続き動作するように、後方互換性を提供します。新しいファイアウォールスクリプトの場合には、Red Hat は **nftables** を使用することを推奨します。

重要

異なるファイアウォールサービスが相互に影響することを回避するには、RHEL ホストでそのうちの 1 つだけを実行し、他のサービスを無効にします。

1.1.2. ゾーン

firewalld は、インターフェースに追加する信頼レベルと、そのネットワークのトラフィックに従って、複数のネットワークを複数のゾーンに分類できます。接続は、1つのゾーンにしか指定できませんが、ゾーンは多くのネットワーク接続に使用できます。

NetworkManager は、**firewalld** にインターフェースのゾーンを通知します。以下を使用して、ゾーンをインターフェースに割り当てることができます。

- **NetworkManager**
- **firewall-config** ツール
- **firewall-cmd** コマンドラインツール
- RHEL Web コンソール

後者の 3 つは、適切な **NetworkManager** 設定ファイルの編集のみを行います。Web コンソールを使用してインターフェースのゾーンを変更する (**firewall-cmd** または **firewall-config**) と、リクエストが **NetworkManager** に転送され、**firewalld** では処理されません。

事前定義したゾーンは **/usr/lib/firewalld/zones/** ディレクトリーに保存され、利用可能なネットワークインターフェースに即座に適用されます。このファイルは、修正しないと **/etc/firewalld/zones/** ディレクトリーにコピーされません。事前定義したゾーンのデフォルト設定は以下のようになります。

block

IPv4 の場合は icmp-host-prohibited メッセージ、 **IPv6** の場合は icmp6-adm-prohibited メッセージで、すべての着信ネットワーク接続が拒否されます。システムで開始したネットワーク接続のみが可能です。

dmz

公開アクセスは可能ですが、内部ネットワークへのアクセスに制限がある非武装地帯にあるコンピューター向けです。選択した着信接続のみが許可されます。

drop

着信ネットワークパケットは、通知なしで遮断されます。発信ネットワーク接続だけが可能です。

external

マスカレードをルーター用に特別に有効にした外部ネットワークでの使用向けです。自分のコンピューターを保護するため、ネットワーク上の他のコンピューターを信頼しません。選択した着信接続のみが許可されます。

home

そのネットワークでその他のコンピューターをほぼ信頼できる自宅での使用向けです。選択した着信接続のみが許可されます。

internal

そのネットワークでその他のコンピューターをほぼ信頼できる内部ネットワーク向けです。選択した着信接続のみが許可されます。

public

そのネットワークでその他のコンピューターを信頼できないパブリックエリア向けです。選択した着信接続のみが許可されます。

trusted

すべてのネットワーク接続が許可されます。

work

そのネットワークで、その他のコンピューターをほぼ信頼できる職場での使用向けです。選択した着信接続のみが許可されます。

このゾーンのいずれかを **デフォルト** ゾーンに設定できます。インターフェース接続を **NetworkManager** に追加すると、デフォルトゾーンに割り当てられます。**firewalld** のデフォルトゾーンは、インストール時に **public** ゾーンに設定されます。デフォルトゾーンは変更できます。

注記

ネットワークゾーン名は、分かりやすく、ユーザーが妥当な決定をすばやく下せるような名前が付けられています。セキュリティー問題を回避するために、ニーズおよびリスク評価に合わせて、デフォルトゾーンの設定の見直しを行ったり、不要なサービスを無効にしてください。

関連情報

- **firewalld.zone(5)** の man ページ

1.1.3. 事前定義サービス

サービスが、ローカルポート、プロトコル、ソースポート、宛先、そしてサービスが有効になると自動的に読み込まれるファイアウォールのヘルパーモジュールの一覧を指す場合があります。サービスを使用すると、ポートのオープン、プロトコルの定義、パケット転送の有効化などを 1 つ 1 つ行うのではなく、1 回のステップで定義できます。

サービス設定オプションと、一般的なファイル情報は、man ページの **firewalld.service(5)** で説明されています。サービスは、個々の XML 設定ファイルを使用して指定し、名前は、**service-name.xml** のような形式になります。プロトコル名は、**firewalld** のサービス名またはアプリケーション名よりも優先されます。

サービスは、グラフィカルな **firewall-config** ツールと、**firewall-cmd** および **firewall-offline-cmd** を使用して追加または削除できます。

または、**/etc/firewalld/services/** ディレクトリーの XML ファイルを変更できます。ユーザーがサービスを追加または変更しないと、**/etc/firewalld/services/** には、対応する XML ファイルが記載されません。**/usr/lib/firewalld/services/** ディレクトリーのファイルは、サービスを追加または変更する場合にテンプレートとして使用できます。

関連情報

- **firewalld.service(5)** の man ページ

1.1.4. firewalld の起動

手順

1. **firewalld** を開始するには、**root** で次のコマンドを実行します。

```
# systemctl unmask firewalld
# systemctl start firewalld
```

2. システムの起動時に **firewalld** を自動的に起動するように設定するには、**root** で次のコマンドを実行します。

```
# systemctl enable firewalld
```

1.1.5. firewalld の停止

手順

1. **firewalld** を停止するには、**root** で次のコマンドを実行します。

```
# systemctl stop firewalld
```

2. システムの起動時に **firewalld** を自動的に起動しないように設定するには、次のコマンドを実行します。

```
# systemctl disable firewalld
```

3. **firewalld D-Bus** インターフェースにアクセスして firewalld を起動していないこと、そしてその他のサービスが **firewalld** を求めているかどうかを確認するには、次のコマンドを実行します。

```
# systemctl mask firewalld
```

1.1.6. 永続的な firewalld 設定の確認

firewalld 設定ファイルを手動で編集した後など、特定の状況では、変更が正しいことを管理者が確認します。本セクションでは、**firewalld** サービスの永続的な設定を確認する方法を説明します。

前提条件

- **firewalld** サービスが実行している。

手順

1. **firewalld** サービスの永続的な設定を確認します。

```
# firewall-cmd --check-config
success
```

永続的な設定が有効になると、コマンドが **success** を返します。その他の場合は、以下のような詳細で、コマンドがエラーを返します。

```
# firewall-cmd --check-config
Error: INVALID_PROTOCOL: 'public.xml': 'tcpx' not from {'tcp'|'udp'|'sctp'|'dccp'}
```

1.2. FIREWALLD の現在の状況および設定の表示

本セクションでは、**firewalld** の現在のステータス、許可されるサービス、および現在の設定を表示する方法を説明します。

1.2.1. firewalld の現在の状況の表示

ファイアウォールサービス **firewalld** は、システムにデフォルトでインストールされています。CLI インターフェース **firewalld** を使用して、サービスが実行していることを確認します。

手順

1. サービスの状況を表示するには、次のコマンドを実行します。

```
# firewall-cmd --state
```

2. サービスの状況の詳細は、**systemctl status** サブコマンドを実行します。

```
# systemctl status firewalld
firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor pr
   Active: active (running) since Mon 2017-12-18 16:05:15 CET; 50min ago
     Docs: man:firewalld(1)
 Main PID: 705 (firewalld)
    Tasks: 2 (limit: 4915)
   CGroup: /system.slice/firewalld.service
           └─705 /usr/bin/python3 -Es /usr/sbin/firewalld --nofork --nopid
```

1.2.2. GUI を使用して許可されるサービスの表示

グラフィカルの **firewall-config** ツールを使用してサービスの一覧を表示する場合は、**Super** キーを押してアクティビティーの概要を開き、**firewall** と入力して **Enter** を押します。**firewall-config** ツールが表示されます。**Services** タブの下にサービスの一覧が表示されます。

グラフィカルなファイアウォール設定ツールは、コマンドラインを使用して起動できます。

前提条件

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

手順

- コマンドラインを使用してグラフィカルなファイアウォール設定ツールを起動するには、次のコマンドを実行します。

```
$ firewall-config
```

Firewall Configuration ウィンドウが開きます。このコマンドは通常のユーザーとして実行できますが、監理者パスワードが求められる場合もあります。

1.2.3. CLI を使用した firewalld 設定の表示

CLI クライアントで、現在のファイアウォール設定を、複数の方法で表示できます。**--list-all** オプションは、**firewalld** 設定の完全概要を表示します。

firewalld は、ゾーンを使用してトラフィックを管理します。**--zone** オプションでゾーンを指定しないと、コマンドは、アクティブネットワークインターフェースおよび接続に割り当てたデフォルトゾーンに対して有効になります。

手順

- デフォルトゾーンに関連する情報をすべて表示するには、次のコマンドを実行します。

```
# firewall-cmd --list-all
```

```
public
  target: default
  icmp-block-inversion: no
  interfaces:
  sources:
  services: ssh dhcpv6-client
  ports:
  protocols:
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
```

- 設定を表示するゾーンを指定するには、たとえば、**--zone=zone-name** 引数を **firewall-cmd --list-all** コマンドに指定します。

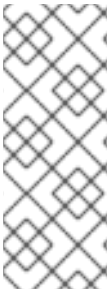
```
# firewall-cmd --list-all --zone=home
home
  target: default
  icmp-block-inversion: no
  interfaces:
  sources:
  services: ssh mdns samba-client dhcpv6-client
... [trimmed for clarity]
```

- サービス、ポートなど、特定情報の設定を確認するには、特定のオプションを使用します。**firewalld** の man ページか、コマンドの help でオプションの一覧を表示します。

```
# firewall-cmd --help
```

- 現在のゾーンで許可されているサービスを表示するには、次のコマンドを実行します。

```
# firewall-cmd --list-services
ssh dhcpv6-client
```



注記

CLI ツールを使用して一覧表示した特定のサブパートの設定は、解釈が難しいことがしばしばあります。たとえば、**firewalld** で **SSH** サービスを許可し、そのサービスに必要なポート (22) を開くことができます。許可されたサービスを一覧表示すると、一覧には **SSH** サービスが表示されますが、開いているポートを一覧表示しても、何も表示されません。したがって、**--list-all** オプションを使用して、完全な情報を取得することが推奨されます。

1.3. FIREWALLD でネットワークトラフィックの制御

本セクションでは、**firewalld** を使用してネットワークトラフィックを制御する方法を説明します。

1.3.1. 緊急時に CLI を使用してすべてのトラフィックの無効化

システムへの攻撃などの緊急な状態にあるとき、すべてのネットワークトラフィックを無効にし、攻撃を遮断できます。

手順

1. ネットワークトラフィックを直ちに無効にするには、パニックモードをオンにします。

```
# firewall-cmd --panic-on
```

重要

パニックモードを有効にすると、ネットワークトラフィックがすべて停止します。したがって、そのマシンへの物理アクセスがある場合、またはシリアルコンソールを使用してログインする場合に限り使用してください。

2. パニックモードをオフにし、ファイアウォールを永続設定に戻します。パニックモードを無効にするには、以下のコマンドを実行します。

```
# firewall-cmd --panic-off
```

検証

- パニックモードが有効または無効であるかを確認するには、以下のコマンドを実行します。

```
# firewall-cmd --query-panic
```

1.3.2. CLI を使用して事前定義されたサービスでトラフィックの制御

トラフィックを制御する最も簡単な方法は、事前定義したサービスを **firewalld** に追加する方法です。これにより、必要なすべてのポートが開き、**service definition file** に従ってその他の設定が変更されます。

手順

1. サービスが許可されていないことを確認します。

```
# firewall-cmd --list-services
ssh dhcpv6-client
```

2. 事前定義したサービスの一覧を表示します。

```
# firewall-cmd --get-services
RH-Satellite-6 amanda-client amanda-k5-client bacula bacula-client bitcoin bitcoin-rpc
bitcoin-testnet bitcoin-testnet-rpc ceph ceph-mon cfengine condor-collector ctdb dhcp dhcpv6
dhcpv6-client dns docker-registry ...
[trimmed for clarity]
```

3. サービスを、許可されたサービスに追加します。

```
# firewall-cmd --add-service=<service-name>
```

4. 新しい設定を永続化します。

```
# firewall-cmd --runtime-to-permanent
```

1.3.3. GUI を使用して事前定義サービスでトラフィックを制御

この手順では、グラフィカルユーザーインターフェースを使用して、事前定義サービスでネットワークトラフィックを制御する方法を説明します。

前提条件

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

手順

1. 事前定義したサービスまたはカスタマイズしたサービスを有効または無効にするには、以下を行います。
   a. **firewall-config** ツールを起動して、サービスを設定するネットワークゾーンを選択します。
   b. **Zones** タブを選択してから、下の **Services** タブを選択します。
   c. 信頼するサービスのタイプごとにチェックボックスをオンにするか、チェックボックスをオフにして、選択したゾーンのサービスをブロックします。
2. サービスを編集するには、以下を行います。
   a. **firewall-config** ツールを起動します。
   b. **Configuration** メニューから **Permanent** を選択します。**Services** ウィンドウの下部に、その他のアイコンおよびメニューボタンが表示されます。
   c. 設定するサービスを選択します。

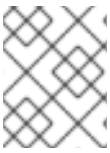
Ports、**Protocols**、**Source Port** のタブでは、選択したサービスのポート、プロトコル、およびソースポートの追加、変更、ならびに削除が可能です。モジュールタブは、**Netfilter** ヘルパーモジュールの設定を行います。**Destination** タブは、特定の送信先アドレスとインターネットプロトコル (**IPv4** または **IPv6**) へのトラフィックが制限できます。

注記

ランタイム モードでは、サービス設定を変更できません。

1.3.4. 新しいサービスの追加

サービスは、グラフィカルな firewall-config ツールと、**firewall-cmd** および **firewall-offline-cmd** を使用して追加または削除できます。または、**/etc/firewalld/services/** にある XML ファイルを編集できます。ユーザーがサービスを追加または変更しないと、対応する XML ファイルが **/etc/firewalld/services/** に作成されません。**/usr/lib/firewalld/services/** のファイルは、サービスを追加または変更する際にテンプレートとして使用できます。



注記

サービス名は英数字にする必要があります。_ (下線) 文字および **-** (ハイフン) 文字も使用できます。

手順

firewalld がアクティブでない場合に、ターミナルで新しいサービスを追加するには、**firewall-cmd** または **firewall-offline-cmd** を使用します。

1. 新しい、空のサービスを追加するには、次のコマンドを実行します。

```
$ firewall-cmd --new-service=service-name --permanent
```

2. ローカルファイルを使用して新規サービスを追加するには、次のコマンドを使用します。

```
$ firewall-cmd --new-service-from-file=service-name.xml --permanent
```

追加オプション **--name=service-name** を指定して、サービス名を変更できます。

3. サービス設定を変更すると、直ちにサービスの更新コピーが **/etc/firewalld/services/** に作成できます。
root で次のコマンドを実行して、サービスを手動でコピーします。

```
# cp /usr/lib/firewalld/services/service-name.xml /etc/firewalld/services/service-name.xml
```

firewalld は、最初に **/usr/lib/firewalld/services** のファイルを読み込みます。ファイルは **/etc/firewalld/services** に置かれ、そのファイルが有効な場合は、**/usr/lib/firewalld/services** で一致するファイルを上書きします。**/usr/lib/firewalld/services** で上書きしたファイルは、**/etc/firewalld/services** で一致するファイルが削除されるとすぐに、もしくはサービスのデフォルトを読み込むように **firewalld** が求められた場合に使用されます。これに該当するのは永続環境のみです。ランタイム環境でフォールバックさせるには、再読み込みが必要です。

1.3.5. GUI を使用してポートを開く

特定のポートへのファイアウォールを通過するトラフィックを許可する場合は、GUI でポートを開くことができます。

前提条件

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

手順

1. firewall-config ツールを起動して、設定を変更するネットワークゾーンを選択します。
2. 右側の **Ports** タブを選択し、**Add** ボタンをクリックします。**Port and Protocol** ウィンドウが開きます。
3. 許可するポート番号またはポートの範囲を入力します。
4. リストから **tcp** または **udp** を選択します。

1.3.6. GUI を使用してプロトコルを使用したトラフィックの制御

特定のプロトコルを使用してファイアウォールを経由したトラフィックを許可するには、GUI を使用できます。

前提条件

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

手順

1. **firewall-config** ツールを起動して、設定を変更するネットワークゾーンを選択します。
2. 右側で **Protocols** タブを選択し、**Add** ボタンをクリックします。**Protocol** ウィンドウが開きます。
3. リストからプロトコルを選択するか、**Other Protocol** チェックボックスを選択し、そのフィールドにプロトコルを入力します。

1.3.7. GUI を使用してソースポートを開く

特定ポートからファイアウォールを経由したトラフィックを許可するには、GUI を使用できます。

前提条件

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

手順

1. firewall-config ツールを起動し、設定を変更するネットワークゾーンを選択します。
2. 右側の **Source Port** タブを選択し、**Add** ボタンをクリックします。**Source Port** ウィンドウが開きます。
3. 許可するポート番号またはポートの範囲を入力します。リストから **tcp** または **udp** を選択します。

1.4. CLI を使用したポートの制御

ポートは、オペレーティングシステムが、ネットワークトラフィックを受信し、区別し、システムサービスに従って転送する論理デバイスです。これは、通常、ポートをリッスンするデーモンにより示されますが、このポートに入るトラフィックを待ちます。

通常、システムサービスは、サービスに予約されている標準ポートでリッスンします。**httpd** デーモンは、たとえば、ポート 80 をリッスンします。ただし、デフォルトでは、システム管理者は、セキュリティーを強化するため、またはその他の理由により、別のポートをリッスンするようにデーモンを設定します。

1.4.1. ポートを開く

開かれたポートを介して、システムが外部からアクセスできます。これはセキュリティーリスクでもあります。一般的に、ポートを閉じたままにし、特定サービスに要求される場合に限り開きます。

手順

現在のゾーンで開かれたポートの一覧を表示するには、以下を行います。

1. 許可されているポートの一覧を表示します。

```
# firewall-cmd --list-ports
```

2. 許可されているポートにポートを追加して、着信トラフィックに対してそのポートを開きます。

```
# firewall-cmd --add-port=port-number/port-type
```

ポートタイプは、**tcp**、**udp**、**sctp**、または **dccp** になります。このタイプは、ネットワーク接続の種類と一致させる必要があります。

3. 新しい設定を永続化します。

```
# firewall-cmd --runtime-to-permanent
```

ポートタイプは、**tcp**、**udp**、**sctp**、または **dccp** になります。このタイプは、ネットワーク接続の種類と一致させる必要があります。

1.4.2. ポートを閉じる

開しているポートが必要なくなった場合に、**firewalld** のポートを閉じます。ポートをそのままにするとセキュリティーリスクとなるため、使用されなくなったらすぐに不要なポートを閉じることが強く推奨されます。

手順

ポートを閉じるには、許可されているポートの一覧からそれを削除します。

1. 許可されているポートの一覧を表示します。

```
# firewall-cmd --list-ports
```

警告

このコマンドにより、ポートとして開かれているポートのみが表示されます。サービスとして開いているポートは表示されません。したがって、**--list-ports** ではなく **--list-all** オプションの使用を検討してください。

2. 「許可されているポート」からポートを削除し、着信トラフィックに対して閉じます。

```
# firewall-cmd --remove-port=port-number/port-type
```

3. 新しい設定を永続化します。

```
# firewall-cmd --runtime-to-permanent
```

1.5. システムロールを使用したポートの設定

Red Hat Enterprise Linux(RHEL) **firewalld** システムロールを使用して、着信トラフィックに対してローカルファイアウォールでポートを開くか、または閉じて、再起動後に新しい設定を永続化できます。この例では、**HTTPS** サービスの着信トラフィックを許可するデフォルトゾーンを設定する方法を説明します。

Ansible コントロールノードで以下の手順を実行します。

前提条件

- **firewalld** システムロールを使用して設定する 1 つ以上の **管理ノード** へのアクセスおよびパーミッション。
- **コントロールノード** (このシステムから Red Hat Ansible Core は他のシステムを設定) へのアクセスおよびパーミッション。
- **ansible-core** パッケージおよび **rhel-system-roles** パッケージが制御ノードにインストールされている。
- Playbook の実行時に **root** 以外のリモートユーザーを使用する場合は、管理ノードで適切な **sudo** パーミッションが付与される。
- ホストは NetworkManager を使用してネットワークを設定している。

手順

1. Playbook の命令を実行するホストのインベントリーがまだ指定されていない場合は、そのホストの IP または名前を Ansible インベントリーファイル **/etc/ansible/hosts** に追加します。

```
node.example.com
```

2. ~/**adding-and-removing-ports.yml** Playbook を以下の内容で作成します。

```
---
- name: Allow incoming HTTPS traffic to the local host
  hosts: node.example.com
  become: true

  tasks:
    - include_role:
        name: linux-system-roles.firewall

      vars:
        firewall:
          - port: 443/tcp
            service: http
            state: enabled
            runtime: true
            permanent: true
```

permanent: true オプションを使用すると、再起動後も新しい設定が維持されます。

3. Playbook を実行します。

- **root** ユーザーとして管理対象ホストに接続するには、次のコマンドを実行します。

```
# ansible-playbook -u root ~/adding-and-removing-ports.yml
```

- 管理ホストにユーザーとして接続するには、次のコマンドを実行します。

```
# ansible-playbook -u user_name --ask-become-pass ~/adding-and-removing-ports.yml
```

--ask-become-pass オプションは、**ansible-playbook** コマンドが **-u user_name** オプションで定義したユーザーの **sudo** パスワードを要求するようにします。

-u user_name オプションを指定しないと、**ansible-playbook** は、コントロールノードに現在ログインしているユーザーとして管理ホストに接続します。

検証

1. 管理ノードに接続します。

```
$ ssh user_name@node.example.com
```

2. **HTTPS** サービスに関連付けられた **443/tcp** ポートが開いていることを確認します。

```
$ sudo firewall-cmd --list-ports
443/tcp
```

関連情報

- **/usr/share/ansible/roles/rhel-system-roles.network/README.md**
- **ansible-playbook(1)** の man ページ

1.6. ファイアウォールゾーンでの作業

ゾーンは、着信トラフィックをより透過的に管理する概念を表しています。ゾーンはネットワークインターフェースに接続されているか、ソースアドレスの範囲に割り当てられます。各ゾーンは個別にファイアウォールルールを管理しますが、これにより、複雑なファイアウォール設定を定義してトラフィックに割り当てることができます。

1.6.1. ゾーンの一覧

この手順では、コマンドラインでゾーンの一覧を表示する方法を説明します。

手順

1. システムで利用可能なゾーンを確認するには、次のコマンドを実行します。

```
# firewall-cmd --get-zones
```

firewall-cmd --get-zones コマンドは、システムで利用可能な全てのゾーンを表示し、特定ゾーンの詳細は表示しません。

2. すべてのゾーンで詳細情報を表示する場合は、次のコマンドを実行します。

```
# firewall-cmd --list-all-zones
```

3. 特定ゾーンに関する詳細情報を表示する場合は、次のコマンドを実行します。

```
# firewall-cmd --zone=zone-name --list-all
```

1.6.2. 特定ゾーンに対する firewalld 設定の修正

CLI を使用して事前定義されたサービスでトラフィックの制御 および CLI を使用したポートの制御 では、サービスを追加する方法や、現在のワークゾーンの範囲内でポートを変更する方法について説明します。別のゾーンへのルールの設定が必要になる場合もあります。

手順

- 別のゾーンに指定するには、**--zone=zone-name** オプションを使用します。たとえば、**public** ゾーンで **SSH** サービスを許可するには、次のコマンドを実行します。

```
# firewall-cmd --add-service=ssh --zone=public
```

1.6.3. デフォルトゾーンの変更

システム管理者は、設定ファイルのネットワークインターフェースにゾーンを割り当てます。特定のゾーンに割り当てられないインターフェースは、デフォルトゾーンに割り当てられます。**firewalld** サービスを再起動するたびに、**firewalld** は、デフォルトゾーンの設定を読み込み、それをアクティブにします。

手順

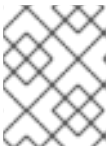
デフォルトゾーンを設定するには、以下を行います。

1. 現在のデフォルトゾーンを表示します。

```
# firewall-cmd --get-default-zone
```

2. 新しいデフォルトゾーンを設定します。

```
# firewall-cmd --set-default-zone zone-name
```



注記

この手順では、**--permanent** オプションを使用しなくても、設定は永続化します。

1.6.4. ゾーンへのネットワークインターフェースの割り当て

複数のゾーンに複数のルールセットを定義して、使用されているインターフェースのゾーンを変更することで、迅速に設定を変更できます。各インターフェイスに特定のゾーンを設定して、そのゾーンを通過するトラフィックを設定できます。

手順

特定インターフェースにゾーンを割り当てるには、以下を行います。

1. アクティブゾーン、およびそのゾーンに割り当てられているインターフェースを一覧表示します。

```
# firewall-cmd --get-active-zones
```

2. 別のゾーンにインターフェースを割り当てます。

```
# firewall-cmd --zone=zone_name --change-interface=interface_name --permanent
```

1.6.5. nmcliを使用して接続にゾーンを割り当て

この手順では、**nmcli** ユーティリティーを使用して、**firewalld** ゾーンを **NetworkManager** 接続に追加する方法を説明します。

手順

1. ゾーンを **NetworkManager** 接続プロファイルに割り当てます。

```
# nmcli connection modify profile connection.zone zone_name
```

2. 接続をアクティベートします。

```
# nmcli connection up profile
```

1.6.6. ifcfg ファイルでゾーンをネットワーク接続に手動で割り当て

NetworkManager で接続を管理する場合は、NetworkManager が使用するゾーンを認識する必要があります。すべてのネットワーク接続にゾーンを指定できます。これにより、ポータブルデバイスを使用したコンピューターの場所に従って、様々なファイアウォールを柔軟に設定できるようになります。したがって、ゾーンおよび設定には、会社または自宅など、様々な場所を指定できます。

手順

- 接続のゾーンを設定するには、**/etc/sysconfig/network-scripts/ifcfg-connection_name** ファイルを変更して、この接続にゾーンを割り当てる行を追加します。

```
ZONE=zone_name
```

1.6.7. 新しいゾーンの作成

カスタムゾーンを使用するには、新しいゾーンを作成したり、事前定義したゾーンなどを使用したりします。新しいゾーンには **--permanent** オプションが必要となり、このオプションがなければコマンドは動作しません。

手順

1. 新しいゾーンを作成します。

```
# firewall-cmd --permanent --new-zone=zone-name
```

2. 作成したゾーンが永続設定に追加されたかどうかを確認します。

```
# firewall-cmd --get-zones
```

3. 新しい設定を永続化します。

```
# firewall-cmd --runtime-to-permanent
```

1.6.8. ゾーンの設定ファイル

また、**ゾーンの設定ファイル** を使用してゾーンを作成できます。このアプローチは、新しいゾーンを作成する必要がある場合に、別のゾーンの設定を変更して利用する場合に便利です。

firewalld ゾーン設定ファイルには、ゾーンに対する情報があります。これは、XML ファイル形式で、ゾーンの説明、サービス、ポート、プロトコル、icmp-block、マスカレード、転送ポート、およびリッチ言語ルールです。ファイル名は **zone-name.xml** となります。**zone-name** の長さは 17 文字に制限されます。ゾーンの設定ファイルは、**/usr/lib/firewalld/zones/** ディレクトリーおよび **/etc/firewalld/zones/** ディレクトリーに置かれています。

以下の例は、**TCP** プロトコルまたは **UDP** プロトコルの両方に、1 つのサービス (**SSH**) および 1 つのポート範囲を許可する設定を示します。

```
<?xml version="1.0" encoding="utf-8"?>
<zone>
  <short>My Zone</short>
  <description>Here you can describe the characteristic features of the zone.</description>
  <service name="ssh"/>
  <port protocol="udp" port="1025-65535"/>
  <port protocol="tcp" port="1025-65535"/>
</zone>
```

そのゾーンの設定を変更するには、セクションを追加または削除して、ポート、転送ポート、サービスなどを追加します。

関連情報

- **firewalld.zone** man ページ

1.6.9. 着信トラフィックにデフォルトの動作を設定するゾーンターゲットの使用

すべてのゾーンに対して、特に指定されていない着信トラフィックを処理するデフォルト動作を設定できます。そのような動作は、ゾーンのターゲットを設定することで定義されます。4 つのオプションがあります。

- **ACCEPT**: 指定したルールで許可されていないパケットを除いた、すべての着信パケットを許可します。
- **REJECT**: 指定したルールで許可されているパケット以外の着信パケットをすべて拒否します。**firewalld** がパケットを拒否すると、送信元マシンに拒否について通知されます。
- **DROP**: 指定したルールで許可されているパケット以外の着信パケットをすべて破棄します。**firewalld** がパケットを破棄すると、ソースマシンにパケット破棄の通知がされません。
- **default**:**REJECT** と似ていますが、特定のシナリオで特別な意味を持ちます。詳細は、**firewall-cmd(1)** man ページの **適応およびクエリーゾーンとポリシーのオプション** セクションを参照してください。

手順

ゾーンにターゲットを設定するには、以下を行います。

1. 特定ゾーンに対する情報を一覧表示して、デフォルトゾーンを確認します。

```
# firewall-cmd --zone=zone-name --list-all
```

2. ゾーンに新しいターゲットを設定します。

```
# firewall-cmd --permanent --zone=zone-name --set-target=
<default|ACCEPT|REJECT|DROP>
```

関連情報

- man ページの **firewall-cmd (1)**

1.7. ゾーンを使用し、ソースに応じた着信トラフィックの管理

ゾーンを使用して、そのソースに基づいて着信トラフィックを管理するゾーンを使用できます。これにより、着信トラフィックを分類し、複数のゾーンに向け、トラフィックにより到達できるサービスを許可または拒否できます。

ソースをゾーンに追加する場合は、ゾーンがアクティブになり、そのソースからの着信トラフィックは、それを介して行われます。各ゾーンに異なる設定を指定できますが、それは指定したソースから順次トラフィックに適用されます。ネットワークインターフェースが 1 つしかない場合でも、複数のゾーンを使用できます。

1.7.1. ソースの追加

着信トラフィックを特定のゾーンに転送する場合は、そのゾーンにソースを追加します。ソースは、CIDR (Classless Inter-domain Routing) 表記法の IP アドレスまたは IP マスクになります。

注記

ネットワーク範囲が重複している複数のゾーンを追加する場合は、ゾーン名で順序付けされ、最初のゾーンのみが考慮されます。

- 現在のゾーンにソースを設定するには、次のコマンドを実行します。

```
# firewall-cmd --add-source=<source>
```

- 特定ゾーンのソース IP アドレスを設定するには、次のコマンドを実行します。

```
# firewall-cmd --zone=zone-name --add-source=<source>
```

以下の手順は、**信頼される** ゾーンで **192.168.2.15** からのすべての着信トラフィックを許可します。

手順

1. すべての利用可能なゾーンを一覧表示します。

```
# firewall-cmd --get-zones
```

2. 永続化モードで、信頼ゾーンにソース IP を追加します。

```
# firewall-cmd --zone=trusted --add-source=192.168.2.15
```

3. 新しい設定を永続化します。

```
# firewall-cmd --runtime-to-permanent
```

1.7.2. ソースの削除

ゾーンからソースを削除すると、そのゾーンからのトラフィックを遮断します。

手順

1. 必要なゾーンに対して許可されているソースの一覧を表示します。

```
# firewall-cmd --zone=zone-name --list-sources
```

2. ゾーンからソースを永続的に削除します。

```
# firewall-cmd --zone=zone-name --remove-source=<source>
```

3. 新しい設定を永続化します。

```
# firewall-cmd --runtime-to-permanent
```

1.7.3. ソースポートの追加

発信源となるポートに基づいたトラフィックの分類を有効にするには、**--add-source-port** オプションを使用してソースポートを指定します。**--add-source** オプションと組み合わせて、トラフィックを特定の IP アドレスまたは IP 範囲に制限できます。

手順

- ソースポートを追加するには、次のコマンドを実行します。

```
# firewall-cmd --zone=zone-name --add-source-port=<port-name>/<tcp|udp|sctp|dccp>
```

1.7.4. ソースポートの削除

ソースポートを削除して、送信元ポートに基づいてトラフィックの分類を無効にします。

手順

- ソースポートを削除するには、次のコマンドを実行します。

```
# firewall-cmd --zone=zone-name --remove-source-port=<port-name>/<tcp|udp|sctp|dccp>
```

1.7.5. ゾーンおよびソースを使用して特定ドメインのみに対してサービスの許可

特定のネットワークからのトラフィックを許可して、マシンのサービスを使用するには、ゾーンおよびソースを使用します。以下の手順では、他のトラフィックがブロックされている間に **192.0.2.0/24** ネットワークからの HTTP トラフィックのみを許可します。

警告

このシナリオを設定する場合は、**default** のターゲットを持つゾーンを使用します。**192.0.2.0/24** からのトラフィックではネットワーク接続がすべて許可されるため、ターゲットが **ACCEPT** に設定されたゾーンを使用することは、セキュリティー上のリスクになります。

手順

1. すべての利用可能なゾーンを一覧表示します。

```
# firewall-cmd --get-zones
block dmz drop external home internal public trusted work
```

2. IP 範囲を **internal** ゾーンに追加し、ソースから発信されるトラフィックをゾーン経由でルーティングします。

```
# firewall-cmd --zone=internal --add-source=192.0.2.0/24
```

3. **http** サービスを **internal** ゾーンに追加します。

```
# firewall-cmd --zone=internal --add-service=http
```

4. 新しい設定を永続化します。

```
# firewall-cmd --runtime-to-permanent
```

検証

- **internal** ゾーンがアクティブで、サービスが許可されていることを確認します。

```
# firewall-cmd --zone=internal --list-all
internal (active)
  target: default
  icmp-block-inversion: no
  interfaces:
  sources: 192.0.2.0/24
  services: cockpit dhcpv6-client mdns samba-client ssh http
  ...
```

関連情報

- **firewalld.zones(5)** の man ページ

1.8. ゾーン間で転送されるトラフィックのフィルタリング

ポリシーオブジェクトを使用すると、ユーザーはポリシーで同様のパーミッションを必要とする異なるアイデンティティーをグループ化できます。トラフィックの方向に応じてポリシーを適用できます。

ポリシーオブジェクトpolicy objects 機能は、firewalld で正引きフィルターと出力フィルターを提供します。ここでは、firewalld を使用して、異なるゾーンのトラフィックをフィルタリングし、ローカルでホストされている仮想マシンへのアクセスを許可してホストに接続する方法を説明します。

1.8.1. ポリシーオブジェクトとゾーンの関係

ポリシーオブジェクトを使用すると、サービス、ポート、リッチルールなどの firewalld のプリミティブをポリシーに割り当てることができます。ポリシーオブジェクトは、ステートフルおよび一方向の方法でゾーン間を通過するトラフィックに適用することができます。

```
# firewall-cmd --permanent --new-policy myOutputPolicy

# firewall-cmd --permanent --policy myOutputPolicy --add-ingress-zone HOST

# firewall-cmd --permanent --policy myOutputPolicy --add-egress-zone ANY
```

HOST および **ANY** は、入出力ゾーンリストで使用されるシンボリックゾーンです。

- **HOST** シンボリックゾーンは、firewalld を実行しているホストから発信されるトラフィック、またはホストへの宛先を持つトラフィックのポリシーを許可します。
- **ANY** シンボリックゾーンは、現行および将来のすべてのゾーンにポリシーを適用します。**ANY**シンボリックゾーンは、すべてのゾーンのワイルドカードとして機能します。

1.8.2. 優先度を使用したポリシーのソート

同じトラフィックセットに複数のポリシーを適用できるため、優先度を使用して、適用される可能性のあるポリシーの優先順位を作成する必要があります。

ポリシーをソートする優先度を設定するには、次のコマンドを実行します。

```
# firewall-cmd --permanent --policy mypolicy --set-priority -500
```

この例では、**-500** の優先度は低くなりますが、優先度は高くなります。したがって、-500 は、-100 より前に実行されます。優先度の高い値は、低い値よりも優先されます。

ポリシーの優先度には、以下のルールが適用されます。

- 負の優先度を持つポリシーは、ゾーンのルールの前に適用されます。
- 正の優先度を持つポリシーは、ゾーンのルールの後に適用されます。
- 優先度 0 は予約されているため、使用できません。

1.8.3. ポリシーオブジェクトを使用した、ローカルでホストされているコンテナーと、ホストに物理的に接続されているネットワークとの間でのトラフィックのフィルタリング

ポリシーオブジェクト機能を使用すると、コンテナーと仮想マシンのトラフィックをフィルターにかけることができます。

手順

1. 新しいポリシーを作成します。

```
# firewall-cmd --permanent --new-policy podmanToHost
```

2. すべてのトラフィックをブロックします。

```
# firewall-cmd --permanent --policy podmanToHost --set-target REJECT

# firewall-cmd --permanent --policy podmanToHost --add-service dhcp

# firewall-cmd --permanent --policy podmanToHost --add-service dns
```

注記

Red Hat では、デフォルトでホストへのすべてのトラフィックをブロックしてから、ホストに必要なサービスを選択的に開くことを推奨しています。

3. ポリシーで使用する入力ゾーンを定義します。

```
# firewall-cmd --permanent --policy podmanToHost --add-ingress-zone podman
```

4. ポリシーで使用する出力ゾーンを定義します。

```
# firewall-cmd --permanent --policy podmanToHost --add-egress-zone ANY
```

検証

- ポリシーに関する情報を確認します。

```
# firewall-cmd --info-policy podmanToHost
```

1.8.4. ポリシーオブジェクトのデフォルトターゲットの設定

ポリシーには --set-target オプションを指定できます。以下のターゲットを使用できます。

- **ACCEPT** - パケットを受け入れます。
- **DROP** - 不要なパケットを破棄します。
- **REJECT** - ICMP リプライで不要なパケットを拒否する
- **CONTINUE (デフォルト)** - パケットは、次のポリシーおよびゾーンのルールに従います。

```
# firewall-cmd --permanent --policy mypolicy --set-target CONTINUE
```

検証

- ポリシーに関する情報の確認

```
# firewall-cmd --info-policy mypolicy
```

1.9. FIREWALLD を使用した NAT の設定

firewalld では、以下のネットワークアドレス変換 (NAT) タイプを設定できます。

- マスカレーディング
- ソース NAT (SNAT)
- 宛先 NAT (DNAT)
- リダイレクト

1.9.1. 異なる NAT タイプ: マスカレード、ソース NAT、宛先 NAT、リダイレクト

以下は、さまざまなネットワークアドレス変換 (NAT) タイプになります。

マスカレードおよびソースの NAT (SNAT)

この NAT タイプのいずれかを使用して、パケットのソース IP アドレスを変更します。たとえば、インターネットサービスプロバイダーは、プライベート IP 範囲 (**10.0.0.0/8** など) をルーティングしません。ネットワークでプライベート IP 範囲を使用し、ユーザーがインターネット上のサーバーにアクセスできるようにする必要がある場合は、この範囲のパケットのソース IP アドレスをパブリック IP アドレスにマップします。
マスカレードおよび SNAT の両方は非常に似ています。相違点は次のとおりです。

- マスカレードは、出力インターフェースの IP アドレスを自動的に使用します。したがって、出力インターフェースが動的 IP アドレスを使用する場合は、マスカレードを使用します。
- SNAT は、パケットのソース IP アドレスを指定された IP に設定し、出力インターフェースの IP アドレスを動的に検索しません。そのため、SNAT の方がマスカレードよりも高速です。出力インターフェースが固定 IP アドレスを使用する場合は、SNAT を使用します。

宛先 NAT (DNAT)

この NAT タイプを使用して、着信パケットの宛先アドレスとポートを書き換えます。たとえば、Web サーバーがプライベート IP 範囲の IP アドレスを使用しているため、インターネットから直接アクセスできない場合は、ルーターに DNAT ルールを設定し、着信トラフィックをこのサーバーにリダイレクトできます。

リダイレクト

このタイプは、チェーンフックに応じてパケットをローカルマシンにリダイレクトする DNAT の特殊なケースです。たとえば、サービスが標準ポートとは異なるポートで実行する場合は、標準ポートからこの特定のポートに着信トラフィックをリダイレクトすることができます。

1.9.2. IP アドレスのマスカレードの設定

以下の手順では、システムで IP マスカレードを有効にする方法を説明します。IP マスカレードは、インターネットにアクセスする際にゲートウェイの向こう側にある個々のマシンを隠します。

手順

1. **external** ゾーンなどで IP マスカレーディングが有効かどうかを確認するには、 **root** で次のコマンドを実行します。

```
# firewall-cmd --zone=external --query-masquerade
```

このコマンドでは、有効な場合は **yes** と出力され、終了ステータスは **0** になります。無効の場合は **no** と出力され、終了ステータスは **1** になります。**zone** を省略すると、デフォルトのゾーンが使用されます。

2. IP マスカレードを有効にするには、**root** で次のコマンドを実行します。

```
# firewall-cmd --zone=external --add-masquerade
```

3. この設定を永続化するには、**--permanent** オプションをコマンドに渡します。

4. IP マスカレードを無効にするには、**root** で次のコマンドを実行します。

```
# firewall-cmd --zone=external --remove-masquerade
```

この設定を永続化するには、**--permanent** をコマンドラインに渡します。

1.10. ポート転送

この方法を使用するポートのリダイレクトは、IPv4 ベースのトラフィックでのみ機能します。IPv6 リダイレクト設定には、リッチルールを使用する必要があります。

外部システムにリダイレクトするには、マスカレードを有効にする必要があります。詳細は、「IP アドレスのマスカレードの設定」を参照してください。

注記

ローカル転送を設定しているホストからリダイレクトされたポートを介してサービスにアクセスすることはできません。

1.10.1. リダイレクトするポートの追加

firewalld を使用して、システムで特定のポートを到達するための着信トラフィックが、選択した別の内部ポート、または別のマシンの外部ポートに配信されるようにポートのリダイレクトを設定できます。

前提条件

- あるポートから別のポートにトラフィックをリダイレクトする前に、パケットが到達するポート、使用されるプロトコル、リダイレクト先を確認しておく必要があります。

手順

1. ポートを別のポートにリダイレクトする場合は、次のコマンドを実行します。

```
# firewall-cmd --add-forward-port=port=port-number:proto=tcp|udp|sctp|dccp:toport=port-number
```

2. 別の IP アドレスで、別のポートにポートをリダイレクトする場合は、次のコマンドを実行します。

   a. 転送するポートを追加します。

```
# firewall-cmd --add-forward-port=port=port-number:proto=tcp|udp:toport=port-number:toaddr=IP
```

b. マスカレードを有効にします。

```
# firewall-cmd --add-masquerade
```

1.10.2. 同一マシンで TCP ポート 80 からポート 88 へのリダイレクト

TCP ポート 80 をポート 88 にリダイレクトするには、以下の手順に従います。

手順

1. TCP トラフィックに対して、ポート 80 からポート 88 へリダイレクトします。

```
# firewall-cmd --add-forward-port=port=80:proto=tcp:toport=88
```

2. 新しい設定を永続化します。

```
# firewall-cmd --runtime-to-permanent
```

3. そのポートがリダイレクトされていることを確認します。

```
# firewall-cmd --list-all
```

1.10.3. リダイレクトしているポートの削除

この手順では、リダイレクトしたポートを削除する方法を説明します。

手順

1. リダイレクトしているポートを削除するには、次のコマンドを実行します。

```
# firewall-cmd --remove-forward-port=port=port-number:proto=<tcp|udp>:toport=port-number:toaddr=<IP>
```

2. 別のアドレスにリダイレクトした転送ポートを削除するには、以下を実行します。

   a. 転送したポートを削除するには、以下を行います。

```
# firewall-cmd --remove-forward-port=port=port-number:proto=<tcp|udp>:toport=port-number:toaddr=<IP>
```

   b. マスカレードを無効にするには、次のコマンドを実行します。

```
# firewall-cmd --remove-masquerade
```

1.10.4. 同じマシンで TCP ポート 88 に転送されるポート 80 の削除

この手順では、ポートのリダイレクト設定を削除する方法を説明します。

手順

1. リダイレクトしたポートの一覧を表示します。

```
~]# firewall-cmd --list-forward-ports
port=80:proto=tcp:toport=88:toaddr=
```

2. ファイアウォールからリダイレクトしたポートを削除します。

```
~]# firewall-cmd  --remove-forward-port=port=80:proto=tcp:toport=88:toaddr=
```

3. 新しい設定を永続化します。

```
~]# firewall-cmd --runtime-to-permanent
```

1.11. ICMP リクエストの管理

Internet Control Message Protocol (**ICMP**) は、接続問題 (要求されているサービスが利用できないなど) を示すエラーメッセージと運用情報を送信するために、様々なネットワークデバイスにより使用されているサポート対象のプロトコルです。**ICMP** は、システム間でデータを交換するのに使用されていないため、TCP、UDP などの転送プロトコルとは異なります。

ただし、**ICMP** メッセージ (特に **echo-request** および **echo-reply**) を利用して、ネットワークに関する情報を明らかにし、その情報をさまざまな不正行為に悪用することが可能です。したがって、**firewalld** は、ネットワーク情報を保護するため、**ICMP** リクエストをブロックできます。

1.11.1. ICMP リクエストの一覧表示およびブロック

ICMP リクエストの一覧表示

ICMP リクエストは、**/usr/lib/firewalld/icmptypes/** ディレクトリーにある各 XML ファイルで説明されています。リクエストの説明は、このファイルを参照してください。**firewall-cmd** コマンドは、**ICMP** リクエストの操作を制御します。

- 利用可能な **ICMP** タイプの一覧を表示するには、次のコマンドを実行します。

```
# firewall-cmd --get-icmptypes
```

- **ICMP** リクエストは、IPv4、IPv6、またはその両方のプロトコルで使用できます。**ICMP** リクエストが使用されているプロトコルを表示するには、次のコマンドを実行します。

```
# firewall-cmd --info-icmptype=<icmptype>
```

- **ICMP** リクエストのステータスは、リクエストが現在ブロックされている場合は **yes**、ブロックされていない場合は **no** となります。**ICMP** リクエストが現在ブロックされているかどうかを確認するには、次のコマンドを実行します。

```
# firewall-cmd --query-icmp-block=<icmptype>
```

ICMP リクエストのブロックまたはブロックの解除

サーバーが **ICMP** リクエストをブロックした場合は、通常の情報が提供されません。ただし、情報が全く提供されないというわけではありません。クライアントは、特定の **ICMP** リクエストがブロックされている (拒否されている) 情報を受け取ります。**ICMP** リクエストは、特に IPv6 トラフィックを使用すると、接続問題が発生することがあるため、注意深く検討する必要があります。

- **ICMP** リクエストが現在ブロックされているかどうかを確認するには、次のコマンドを実行します。

```
# firewall-cmd --query-icmp-block=<icmptype>
```

- **ICMP** リクエストをブロックするには、次のコマンドを実行します。

```
# firewall-cmd --add-icmp-block=<icmptype>
```

- **ICMP** リクエストのブロックを削除するには、次のコマンドを実行します。

```
# firewall-cmd --remove-icmp-block=<icmptype>
```

情報を提供せずに ICMP リクエストのブロック

通常、**ICMP** リクエストをブロックすると、ブロックしていることをクライアントは認識します。したがって、ライブの IP アドレスを傍受している潜在的な攻撃者は、IP アドレスがオンラインであることを確認できます。この情報を完全に非表示にするには、**ICMP** リクエストをすべて破棄する必要があります。

- すべての **ICMP** リクエストをブロックして破棄するには、次のコマンドを実行します。
- ゾーンのターゲットを **DROP** に設定します。

```
# firewall-cmd --permanent --set-target=DROP
```

これで、明示的に許可されるトラフィックを除き、**ICMP** リクエストを含むすべてのトラフィックが破棄されます。

特定の **ICMP** リクエストをブロックして破棄し、その他のリクエストを許可するには、以下を行います。

1. ゾーンのターゲットを **DROP** に設定します。

```
# firewall-cmd --permanent --set-target=DROP
```

2. すべての **ICMP** リクエストを一度にブロックする、ICMP ブロックの反転を追加します。

```
# firewall-cmd --add-icmp-block-inversion
```

3. 許可する **ICMP** リクエストに ICMP ブロックを追加する場合は、次のコマンドを実行します。

```
# firewall-cmd --add-icmp-block=<icmptype>
```

4. 新しい設定を永続化します。

```
# firewall-cmd --runtime-to-permanent
```

ブロックの反転 は、**ICMP** リクエストブロックの設定を反転します。そのため、ゾーンのターゲットが **DROP** に変更されたため、ブロックされていないリクエストはすべてブロックされます。ブロックされているリクエストはブロックされません。これは、リクエストのブロックを解除する場合は、ブロックコマンドを使用する必要があることを示しています。

ブロックの反転を、完全許可の設定に戻すには、以下を行います。

1. ゾーンのターゲットを **default** または **ACCEPT** に戻すには、次のコマンドを設定します。

```
# firewall-cmd --permanent --set-target=default
```

2. **ICMP** リクエストに追加したすべてのブロックを削除します。

```
# firewall-cmd --remove-icmp-block=<icmptype>
```

3. **ICMP** ブロックの反転を削除します。

```
# firewall-cmd --remove-icmp-block-inversion
```

4. 新しい設定を永続化します。

```
# firewall-cmd --runtime-to-permanent
```

1.11.2. GUI を使用した ICMP フィルターの設定

- **ICMP** フィルターを有効または無効にするには、**firewall-config** ツールを起動して、フィルターをかけるメッセージのネットワークゾーンを選択します。**ICMP フィルター** タブを選択し、フィルターをかける **ICMP** メッセージの各タイプのチェックボックスを選択します。フィルターを無効にするには、チェックボックスの選択を外します。これは方向ごとに設定され、デフォルトではすべてが許可されます。

- **ICMP フィルター** の反転を有効にするには、右側の **フィルターの反転** チェックボックスをクリックします。マークがついた **ICMP** タイプだけが許可され、その他はすべて拒否されます。DROP ターゲットを使用するゾーンでは破棄されます。

1.12. FIREWALLD を使用した IP セットの設定および制御

firewalld で対応する IP セットタイプの一覧を表示するには、root で次のコマンドを実行します。

```
~]# firewall-cmd --get-ipset-types
hash:ip hash:ip,mark hash:ip,port hash:ip,port,ip hash:ip,port,net hash:mac hash:net hash:net,iface
hash:net,net hash:net,port hash:net,port,net
```

1.12.1. CLI を使用した IP セットオプションの設定

IP セットは、**firewalld** ゾーンでソースとして使用でき、リッチルールでソースとして使用できます。Red Hat Enterprise Linux で推奨される方法は、ダイレクトルールで **firewalld** を使用して作成した IP セットを使用する方法です。

- 永続的な環境で **firewalld** に認識されている IP セットの一覧を表示するには、次のコマンドを **root** で実行します。

```
# firewall-cmd --permanent --get-ipsets
```

- 新しい IP セットを追加するには、永続化環境を使用し、**root** で次のコマンドを実行します。

```
# firewall-cmd --permanent --new-ipset=test --type=hash:net
success
```

上記のコマンドは、名前 **test** とタイプ **hash:net** で、**IPv4** の新しい IP セットを作成します。**IPv6** で使用する IP セットを作成する場合は、**--option=family=inet6** オプションを追加します。ランタイム環境で新しい設定を有効にするには、**firewalld** を再読み込みします。

- **root** で次のコマンドを実行して、新しい IP セットの一覧を表示します。

```
# firewall-cmd --permanent --get-ipsets
test
```

- IP セットの詳細は、**root** で次のコマンドを実行します。

```
# firewall-cmd --permanent --info-ipset=test
test
type: hash:net
options:
entries:
```

この時点では IP セットにエントリーがありません。

- IP セット **test** にエントリーを追加するには、**root** で次のコマンドを実行します。

```
# firewall-cmd --permanent --ipset=test --add-entry=192.168.0.1
success
```

上記のコマンドは、IP アドレス **192.168.0.1** を IP セットに追加します。

- IP セットの現在のエントリーを一覧表示するには、**root** で次のコマンドを実行します。

```
# firewall-cmd --permanent --ipset=test --get-entries
192.168.0.1
```

- IP アドレスの一覧を含むファイルを生成します。以下に例を示します。

```
# cat > iplist.txt <<EOL
192.168.0.2
192.168.0.3
192.168.1.0/24
192.168.2.254
EOL
```

IP セットの IP アドレスの一覧が含まれるファイルには、行ごとにエントリーが含まれている必要があります。ハッシュ、セミコロン、また空の行から始まる行は無視されます。

- **iplist.txt** ファイルからアドレスを追加するには、**root** で次のコマンドを実行します。

```
# firewall-cmd --permanent --ipset=test --add-entries-from-file=iplist.txt
success
```

- 拡張された IP セットのエントリー一覧を表示するには、**root** で次のコマンドを実行します。

```
# firewall-cmd --permanent --ipset=test --get-entries
```

```
192.168.0.1
192.168.0.2
192.168.0.3
192.168.1.0/24
192.168.2.254
```

- IP セットからアドレスを削除し、更新したエントリー一覧を確認するには、**root** で次のコマンドを実行します。

```
# firewall-cmd --permanent --ipset=test --remove-entries-from-file=iplist.txt
success
# firewall-cmd --permanent --ipset=test --get-entries
192.168.0.1
```

- IP セットをゾーンへのソースとして追加し、ゾーンを使用して、IP セットに記載されるアドレスから受信するすべてのトラフィックを処理します。たとえば、IP セットの **test** をソースとして **drop** ゾーンに追加し、IP セットの **test** の一覧に表示されるすべてのエントリーから発信されるパケットをすべて破棄するには、**root** で次のコマンドを実行します。

```
# firewall-cmd --permanent --zone=drop --add-source=ipset:test
success
```

ソースの **ipset:** プレフィックスは、ソースが IP セットで、IP アドレスまたはアドレス範囲ではない **firewalld** を示しています。

IP セットの作成および削除は、永続環境に限定されますが、その他の IP セットオプションは、**--permanent** オプションを使用しないランタイム環境で使用できます。

警告

Red Hat は、**firewalld** を介して管理していない IP セットを使用することは推奨しません。このような IP セットを使用すると、そのセットを参照する永続的なダイレクトルールが必要で、IP セットを作成するカスタムサービスを追加する必要があります。このサービスは、**firewalld** を起動する前に起動する必要があります。先に起動しておかないと、**firewalld** が、このセットを使用してダイレクトルールを追加できません。**/etc/firewalld/direct.xml** ファイルを使用して、永続的なダイレクトルールを追加できます。

1.13. リッチルールの優先度設定

デフォルトでは、リッチルールはルールアクションに基づいて構成されます。たとえば、**許可** ルールよりも **拒否** ルールが優先されます。リッチルールで **priority** パラメーターを使用すると、管理者はリッチルールとその実行順序をきめ細かく制御できます。

1.13.1. priority パラメーターを異なるチェーンにルールを整理する方法

リッチルールの **priority** パラメーターは、**-32768** ~ **32767** の任意の数値に設定でき、値が小さい方が優先されます。

firewalld サービスは、優先度の値に基づいて、ルールを異なるチェーンに整理します。

- 優先度が 0 未満 - ルールは **_pre** 接尾辞が付いたチェーンにリダイレクトされます。
- 優先度が 0 を超える - ルールは **_post** 接尾辞が付いたチェーンにリダイレクトされます。
- 優先度が 0 - アクションに基づいて、ルールは、**_log**、**_deny**、または **_allow** のアクションを使用してチェーンにリダイレクトされます。

このサブチェーンでは、**firewalld** は優先度の値に基づいてルールを分類します。

1.13.2. リッチルールの優先度の設定

この手順では、**priority** パラメーターを使用して、他のルールで許可または拒否されていないすべてのトラフィックをログに記録するリッチルールを作成する方法を説明します。このルールを使用して、予期しないトラフィックにフラグを付けることができます。

手順

1. 優先度が非常に低いルールを追加して、他のルールと一致していないすべてのトラフィックをログに記録します。

```
# firewall-cmd --add-rich-rule='rule priority=32767 log prefix="UNEXPECTED: " limit
value="5/m"'
```

このコマンドでは、ログエントリーの数を、毎分 **5** に制限します。

2. 必要に応じて、前の手順のコマンドで作成した **nftables** ルールを表示します。

```
# nft list chain inet firewalld filter_IN_public_post
table inet firewalld {
  chain filter_IN_public_post {
    log prefix "UNEXPECTED: " limit rate 5/minute
  }
}
```

1.14. ファイアウォールロックダウンの設定

ローカルのアプリケーションやサービスは、**root** で実行していれば、ファイアウォール設定を変更できます (たとえば **libvirt**)。管理者は、この機能を使用してファイアウォール設定をロックし、すべてのアプリケーションでファイアウォール変更を要求できなくするか、ロックダウンの許可リストに追加されたアプリケーションのみがファイアウォール変更を要求できるようにすることが可能になります。ロックダウン設定はデフォルトで無効になっています。これを有効にすると、ローカルのアプリケーションやサービスによるファイアウォールへの望ましくない設定変更を確実に防ぐことができます。

1.14.1. CLI を使用したロックダウンの設定

この手順では、コマンドラインでロックダウンを有効または無効にする方法を説明します。

- ロックダウンが有効になっているかどうかを確認するには、**root** で次のコマンドを使用します。

```
# firewall-cmd --query-lockdown
```

ロックダウンが有効な場合は、**yes** と出力され、終了ステータスは **0** になります。無効の場合は **no** と出力され、終了ステータスは **1** になります。

- ロックダウンを有効にするには、**root** で次のコマンドを実行します。

```
# firewall-cmd --lockdown-on
```

- ロックダウンを無効にするには、**root** で次のコマンドを実行します。

```
# firewall-cmd --lockdown-off
```

1.14.2. CLI を使用したロックダウン許可リストオプションの設定

ロックダウンの許可リストには、コマンド、セキュリティーのコンテキスト、ユーザー、およびユーザー ID を追加できます。許可リストのコマンドエントリーがアスタリスク「*」で終了している場合は、そのコマンドで始まるすべてのコマンドラインが一致することになります。「*」がなければ、コマンドと引数が完全に一致する必要があります。

- ここでのコンテキストは、実行中のアプリケーションやサービスのセキュリティー (SELinux) コンテキストです。実行中のアプリケーションのコンテキストを確認するには、次のコマンドを実行します。

```
$ ps -e --context
```

このコマンドは、実行中のアプリケーションをすべて返します。**grep** ツールを使用して、出力から目的のアプリケーションをパイプ処理します。以下に例を示します。

```
$ ps -e --context | grep example_program
```

- 許可リストにあるコマンドラインの一覧を表示するには、**root** で次のコマンドを実行します。

```
# firewall-cmd --list-lockdown-whitelist-commands
```

- 許可リストに **command** コマンドを追加するには、**root** で次のコマンドを実行します。

```
# firewall-cmd --add-lockdown-whitelist-command='/usr/bin/python3 -Es /usr/bin/command'
```

- 許可リストから **command** コマンドを削除するには、**root** で次のコマンドを実行します。

```
# firewall-cmd --remove-lockdown-whitelist-command='/usr/bin/python3 -Es
/usr/bin/command'
```

- **command** コマンドが許可リストに含まれるかどうかを確認するには、**root** で次のコマンドを実行します。

```
# firewall-cmd --query-lockdown-whitelist-command='/usr/bin/python3 -Es /usr/bin/command'
```

このコマンドでは、含まれる場合は **yes** が出力され、終了ステータスは **0** になります。無効の場合は **no** と出力され、終了ステータスは **1** になります。

- 許可リストにあるセキュリティーコンテキストの一覧を表示するには、**root** で次のコマンドを実行します。

```
# firewall-cmd --list-lockdown-whitelist-contexts
```

- 許可リストに **context** コンテキストを追加するには、**root** で次のコマンドを実行します。

```
# firewall-cmd --add-lockdown-whitelist-context=context
```

- 許可リストから **context** コンテキストを削除するには、**root** で次のコマンドを実行します。

```
# firewall-cmd --remove-lockdown-whitelist-context=context
```

- **context** コンテキストが許可リストに含まれるかどうかを確認するには、**root** で次のコマンドを実行します。

```
# firewall-cmd --query-lockdown-whitelist-context=context
```

含まれる場合は、**yes** と出力され、終了ステータスは **0** になります。含まれない場合は、**no** が出力され、終了ステータスは **1** になります。

- 許可リストにあるユーザー ID すべての一覧を表示するには、**root** で次のコマンドを実行します。

```
# firewall-cmd --list-lockdown-whitelist-uids
```

- 許可リストにユーザー ID (**uid**) を追加するには、**root** で次のコマンドを実行します。

```
# firewall-cmd --add-lockdown-whitelist-uid=uid
```

- 許可リストからユーザー ID (**uid**) を削除するには、**root** で次のコマンドを実行します。

```
# firewall-cmd --remove-lockdown-whitelist-uid=uid
```

- 許可リストにユーザー ID (**uid**) があるかどうかを確認するには、次のコマンドを実行します。

```
$ firewall-cmd --query-lockdown-whitelist-uid=uid
```

含まれる場合は、**yes** と出力され、終了ステータスは **0** になります。含まれない場合は、**no** が出力され、終了ステータスは **1** になります。

- 許可リストにある全ユーザー名の一覧を表示するには、**root** で次のコマンドを実行します。

```
# firewall-cmd --list-lockdown-whitelist-users
```

- 許可リストにユーザー名 (**user**) を追加するには、**root** で次のコマンドを実行します。

```
# firewall-cmd --add-lockdown-whitelist-user=user
```

- 許可リストからユーザー名 (**user**) を削除するには、**root** で次のコマンドを実行します。

```
# firewall-cmd --remove-lockdown-whitelist-user=user
```

- ユーザー名 (**user**) が許可リストに含まれるかどうかを確認するには、次のコマンドを実行します。

```
$ firewall-cmd --query-lockdown-whitelist-user=user
```

含まれる場合は、**yes** と出力され、終了ステータスは **0** になります。含まれない場合は、**no** が出力され、終了ステータスは **1** になります。

1.14.3. 設定ファイルを使用したロックダウンの許可リストオプションの設定

デフォルトの許可リスト設定ファイルには、**NetworkManager** コンテキストと、**libvirt** のデフォルトコンテキストが含まれます。リストには、ユーザー ID (0) もあります。

+ 許可リスト設定ファイルは **/etc/firewalld/** ディレクトリーに保存されます。

```
<?xml version="1.0" encoding="utf-8"?>
 <whitelist>
   <selinux context="system_u:system_r:NetworkManager_t:s0"/>
   <selinux context="system_u:system_r:virtd_t:s0-s0:c0.c1023"/>
   <user id="0"/>
 </whitelist>
```

以下の許可リスト設定ファイルの例では、**firewall-cmd** ユーティリティーのコマンドと、ユーザー ID が **815** である **user** のコマンドをすべて有効にしています。

```
<?xml version="1.0" encoding="utf-8"?>
 <whitelist>
   <command name="/usr/libexec/platform-python -s /bin/firewall-cmd*"/>
   <selinux context="system_u:system_r:NetworkManager_t:s0"/>
   <user id="815"/>
   <user name="user"/>
 </whitelist>
```

この例では、**user id** と **user name** の両方が使用されていますが、実際にはどちらか一方のオプションだけが必要です。Python はインタープリターとしてコマンドラインに追加されています。または、以下のような明確なコマンドも使用できます。

```
/usr/bin/python3 /bin/firewall-cmd --lockdown-on
```

この例では、**--lockdown-on** コマンドだけが許可されます。

Red Hat Enterprise Linux では、すべてのユーティリティーが **/usr/bin/** ディレクトリーに格納されており、**/bin/** ディレクトリーは **/usr/bin/** ディレクトリーへのシンボリックリンクとなります。つまり、**root** で **firewall-cmd** へのパスを実行すると **/bin/firewall-cmd** に対して解決しますが、**/usr/bin/firewall-cmd** が使用できるようになっています。新たなスクリプトは、すべて新しい格納場所を使用する必要があります。ただし、**root** で実行するスクリプトが **/bin/firewall-cmd** へのパスを使用するようになっているのであれば、これまでは **root** 以外のユーザーにのみ使用されていた **/usr/bin/firewall-cmd** パスに加え、このコマンドのパスも許可リストに追加する必要があります。

コマンドの名前属性の最後にある * は、その名前で始まるすべてのコマンドが一致することを意味します。「*」がなければ、コマンドと引数が完全に一致する必要があります。

1.15. FIREWALLD ゾーン内の異なるインターフェースまたはソース間でのトラフィック転送の有効化

ゾーン内転送は、**firewalld** ゾーン内のインターフェースまたはソース間のトラフィック転送を可能にする **firewalld** 機能です。

1.15.1. ゾーン内転送と、デフォルトのターゲットが ACCEPT に設定されているゾーンの違い

ゾーン内転送を有効にすると、1 つの **firewalld** ゾーン内のトラフィックは、あるインターフェースまたはソースから別のインターフェースまたはソースに流れることができます。ゾーンは、インターフェースおよびソースの信頼レベルを指定します。信頼レベルが同じである場合、インターフェースまたはソース間の通信が可能です。

firewalld のデフォルトゾーンでゾーン内転送を有効にすると、現在のデフォルトゾーンに追加されたインターフェースおよびソースにのみ適用されることに注意してください。

firewalld の **trusted** ゾーンは、**ACCEPT** に設定されたデフォルトのターゲットを使用します。このゾーンは、転送されたすべてのトラフィックを受け入れ、ゾーン内転送は適用されません。

他のデフォルトのターゲット値の場合、転送されたトラフィックはデフォルトでドロップされます。これは、信頼済みゾーンを除くすべての標準ゾーンに適用されます。

1.15.2. ゾーン内転送を使用したイーサネットと Wi-Fi ネットワーク間でのトラフィックの転送

ゾーン内転送を使用して、同じ **firewalld** ゾーン内のインターフェースとソース間のトラフィックを転送することができます。たとえば、この機能を使用して、**enp1s0** に接続されたイーサネットネットワークと、**wlp0s20** に接続された Wi-Fi ネットワーク間のトラフィックを転送するには、この機能を使用します。

手順

1. カーネルでパケット転送を有効にします。

```
# echo "net.ipv4.ip_forward=1" > /etc/sysctl.d/95-IPv4-forwarding.conf

# sysctl -p /etc/sysctl.d/95-IPv4-forwarding.conf
```

2. ゾーン内転送を有効にするインターフェースが、**internal** ゾーンと異なるゾーンに割り当てられていないことを確認してください。

```
# firewall-cmd --get-active-zones
```

3. 現在、インターフェースが **internal** 以外のゾーンに割り当てられている場合は、以下のように再割り当てします。

```
# firewall-cmd --zone=internal --change-interface=interface_name --permanent
```

4. **enp1s0** および **wlp0s20** インターフェースを **internal** ゾーンに追加します。

```
# firewall-cmd --zone=internal --add-interface=enp1s0 --add-interface=wlp0s20
```

5. ゾーン内転送を有効にします。

```
# firewall-cmd --zone=internal --add-forward
```

検証

以下の検証手順では、**nmap-ncat** パッケージが両方のホストにインストールされている必要があります。

1. ゾーン転送を有効にしたホストの **enp1s0** インターフェースと同じネットワーク内にあるホストにログインします。
2. **ncat** で echo サービスを起動し、接続をテストします。

```
# ncat -e /usr/bin/cat -l 12345
```

3. **wlp0s20** インターフェースと同じネットワークにあるホストにログインします。
4. **enp1s0** と同じネットワークにあるホスト上で実行している echo サーバーに接続します。

```
# ncat <other host> 12345
```

5. 試しに何かを入力して **Enter** キーを押し、テキストが返送されることを確認します。

関連情報

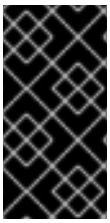
- **firewalld.zones(5)** の man ページ

1.16. ANSIBLE での RHEL システムロールを使用した FIREWALLD 設定の構成

Ansible firewall システムロールを使用すると、一度に複数のクライアントに **firewalld** サービスを設定できます。この解決策は以下のとおりです。

- 効率的な入力設定を持つインターフェースを提供します。
- 目的の **firewalld** パラメーターを 1 か所で保持します。

コントロールノードで **ファイアウォール** ロールを実行すると、システムロールは **firewalld** パラメーターを管理ノードに即座に適用し、再起動後も維持されます。



重要

RHEL チャンネル経由で配信される RHEL システムロールは、RHEL のお客様が、デフォルトの **AppStream** リポジトリーの RPM パッケージとして利用できることに注意してください。RHEL システムロールは、Ansible Automation Hub を介して Ansible サブスクリプションを使用しているお客様のコレクションとしても利用できます。

1.16.1. ファイアウォールの RHEL システムロールの概要

RHEL システムロールは、Ansible 自動化ユーティリティーのコンテンツセットです。このコンテンツは、Ansible 自動化ユーティリティーとともに、複数のシステムをリモートで管理するための一貫した設定インターフェースを提供します。

firewalld サービスの自動設定に、RHEL システムロールからの **rhel-system-roles.firewall** ロールが導入されました。**rhel-system-roles** パッケージには、このシステムロールと参考ドキュメントも含まれます。

firewalld パラメーターを自動化された方法で 1 つ以上のシステムに適用するには、Playbook で **firewall** システムロール変数を使用します。Playbook は、テキストベースの YAML 形式で記述された 1 つ以上のプレイの一覧です。

インベントリーファイルを使用して、Ansible が設定するシステムセットを定義できます。

firewall ロールを使用すると、以下のような異なる **firewalld** パラメーターを設定できます。

- ゾーン。
- パケットが許可されるサービス。
- ポートへのトラフィックアクセスの付与、拒否、または削除。
- ゾーンのポートまたはポート範囲の転送

関連情報

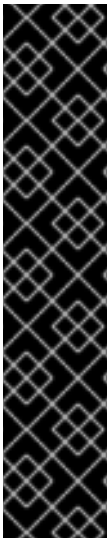
- **/usr/share/doc/rhel-system-roles/firewall/** ディレクトリーの **README.md** ファイルおよび **README.html** ファイル
- Playbook の操作
- インベントリーの構築方法

1.16.2. あるローカルポートから別のローカルポートへの着信トラフィックの転送

rhel-system-roles.firewall ロールを使用すると、複数の管理対象ホストで永続化の影響で **firewalld** パラメーターをリモートで設定できます。

前提条件

- RHEL サブスクリプションで権利がある。**ansible-core** パッケージおよび **rhel-system-roles** パッケージをコントロールノードにインストールしている。
- 管理対象ホストのインベントリーが制御マシンに存在し、Ansible から接続できる。
- 管理対象ホストで Ansible Playbook を実行するパーミッションがある。
- Playbook の実行時に **root** 以外のリモートユーザーを使用する場合は、管理対象ホストで適切な **sudo** パーミッションが付与される。
- インベントリーファイルは、Playbook がアクションを実行するホストを一覧表示します。この手順の Playbook は、グループ **testinservers** のホストで実行します。



重要

RHEL 8.0-8.5 では、別の Ansible リポジトリーへのアクセス権を指定されており、Ansible をベースにする自動化用の Ansible Engine 2.9 が含まれています。Ansible Engine には、 **ansible**、**ansible-playbook**などのコマンドラインユーティリティー、**docker**や**podman**などのコネクター、プラグインとモジュールすべてが含まれています。Ansible Engine を入手してインストールする方法については、Red Hat Ansible Engine をダウンロードしてインストールする方法を参照してください。

RHEL 8.6 以降では、Ansible Core が導入されました（**ansible-core** RPM として提供）。これには、Ansible コマンドラインユーティリティー、コマンド、およびビルトイン Ansible プラグインの小規模なセットが含まれます。App Stream リポジトリーには、**ansible-core** が含まれていますが、サポートの範囲が限定されています。詳細は、RHEL 9App Stream に含まれている ansible-core パッケージのサポート範囲 を確認してください。

手順

1. **~/port_forwarding.yml** ファイルを作成し、以下の内容を追加します。

```
---
- name: Forward incoming traffic on port 8080 to 443
  hosts: testingservers

  tasks:
    - include_role:
        name: rhel-system-roles.firewall

  vars:
    firewall:
      - { forward_port: 8080/tcp;443;, state: enabled, runtime: true, permanent: true }
```

このファイルは Playbook を表し、通常は、**inventory** ファイルから選択した特定の管理対象ホストに対して実行される、**プレイ** とも呼ばれるタスクの順序付きリストが含まれます。この場合、Playbook は管理対象ホストの **testingservers** グループに対して実行されます。

プレイの **hosts** キーは、プレイを実行するホストを指定します。このキーの値または値は、管理対象ホストの個別名または **インベントリー** ファイルで定義されているホストのグループとして指定できます。

tasks セクションには **include_role** キーがあります。このキーは、**vars** セクションで説明されているパラメーターおよび値を設定するシステムロールを指定します。

vars セクションには、**firewall** と呼ばれるロール変数が含まれます。この変数はディクショナリー値のリストで、管理対象ホストの **firewalld** に適用されるパラメーターを指定します。サンプルロールは、ポート 8080 に送信されるトラフィックを 443 ポートに転送します。この設定は即座に有効になり、再起動後も維持されます。

2. 必要に応じて、Playbook の構文が正しいことを確認します。

```
# ansible-playbook --syntax-check ~/port_forwarding.yml

playbook: port_forwarding.yml
```

以下の例では、Playbook の検証が成功したことを示しています。

3. Playbook を実行します。

```
# ansible-playbook ~/port_forwarding.yml
```

検証

- 管理対象ホストで以下を行います。
  - ホストを再起動して、**firewalld** 設定が再起動後に依然として設定されているかどうかを確認します。

```
# reboot
```

  - **firewalld** 設定を表示します。

```
# firewall-cmd --list-forward-ports
```

関連情報

- RHEL システムロールの使用
- **/usr/share/doc/rhel-system-roles/firewall/** ディレクトリーの **README.html** ファイルおよび **README.md** ファイル
- インベントリーの構築
- Ansible の設定
- Playbook の使用
- 変数の使用
- ロール

1.16.3. システムロールを使用したポートの設定

Red Hat Enterprise Linux(RHEL) **firewalld** システムロールを使用して、着信トラフィックに対してローカルファイアウォールでポートを開くか、または閉じて、再起動後に新しい設定を永続化できます。この例では、**HTTPS** サービスの着信トラフィックを許可するデフォルトゾーンを設定する方法を説明します。

Ansible コントロールノードで以下の手順を実行します。

前提条件

- **firewalld** システムロールを使用して設定する 1 つ以上の **管理ノード** へのアクセスおよびパーミッション。
- **コントロールノード** (このシステムから Red Hat Ansible Core は他のシステムを設定) へのアクセスおよびパーミッション。
- **ansible-core** パッケージおよび **rhel-system-roles** パッケージが制御ノードにインストールされている。

- Playbook の実行時に **root** 以外のリモートユーザーを使用する場合は、管理ノードで適切な **sudo** パーミッションが付与される。
- ホストは NetworkManager を使用してネットワークを設定している。

手順

1. Playbook の命令を実行するホストのインベントリーがまだ指定されていない場合は、そのホストの IP または名前を Ansible インベントリーファイル **/etc/ansible/hosts** に追加します。

```
node.example.com
```

2. ~/**adding-and-removing-ports.yml** Playbook を以下の内容で作成します。

```
---
- name: Allow incoming HTTPS traffic to the local host
  hosts: node.example.com
  become: true

  tasks:
    - include_role:
        name: linux-system-roles.firewall

      vars:
        firewall:
          - port: 443/tcp
            service: http
            state: enabled
            runtime: true
            permanent: true
```

permanent: true オプションを使用すると、再起動後も新しい設定が維持されます。

3. Playbook を実行します。
   - **root** ユーザーとして管理対象ホストに接続するには、次のコマンドを実行します。

```
# ansible-playbook -u root ~/adding-and-removing-ports.yml
```

   - 管理ホストにユーザーとして接続するには、次のコマンドを実行します。

```
# ansible-playbook -u user_name --ask-become-pass ~/adding-and-removing-
ports.yml
```

     --ask-become-pass オプションは、**ansible-playbook** コマンドが **-u user_name** オプションで定義したユーザーの **sudo** パスワードを要求するようにします。

-u user_name オプションを指定しないと、**ansible-playbook** は、コントロールノードに現在ログインしているユーザーとして管理ホストに接続します。

検証

1. 管理ノードに接続します。

```
$ ssh user_name@node.example.com
```

2. **HTTPS** サービスに関連付けられた **443/tcp** ポートが開いていることを確認します。

```
$ sudo firewall-cmd --list-ports
443/tcp
```

関連情報

- **/usr/share/ansible/roles/rhel-system-roles.network/README.md**
- **ansible-playbook(1)** の man ページ

1.16.4. firewalld RHEL システムロールを使用した DMZ firewalld ゾーンの設定

システム管理者は、RHEL **firewalld** システムロールを使用して、**enp1s0** インターフェースで **dmz** ゾーンを設定し、ゾーンへの **HTTPS** トラフィックを許可できます。これにより、外部ユーザーが Web サーバーにアクセスできるようにします。

前提条件

- 1 つ以上の **管理対象ノード** (VPN システムロールで設定するシステム) へのアクセスおよびパーミッション。
- **コントロールノード** (このシステムから Red Hat Ansible Core は他のシステムを設定) へのアクセスおよびパーミッション。
- 管理対象ノードを一覧表示するインベントリーファイル。
- **ansible-core** パッケージおよび **rhel-system-roles** パッケージが制御ノードにインストールされている。
- Playbook の実行時に **root** 以外のリモートユーザーを使用する場合は、管理ノードで適切な **sudo** パーミッションが付与される。
- 管理ノードは、**NetworkManager** を使用してネットワークを設定します。

手順

1. **~/configuring-a-dmz-using-the-firewall-system-role.yml** Playbook を以下の内容で作成します。

```
---
- name: Creating a DMZ with access to HTTPS port and masquerading for hosts in DMZ
  hosts: node.example.com
  become: true

  tasks:
    - include_role:
        name: linux-system-roles.firewall

      vars:
        firewall:
          - zone: dmz
            interface: enp1s0
            service: https
```

```
        state: enabled
        runtime: true
        permanent: true
```

2. Playbook を実行します。

   - **root** ユーザーとして管理対象ホストに接続するには、次のコマンドを実行します。

     ```
     $ ansible-playbook -u root ~/configuring-a-dmz-using-the-firewall-system-role.yml
     ```

   - 管理ホストにユーザーとして接続するには、次のコマンドを実行します。

     ```
     $ ansible-playbook -u user_name --ask-become-pass ~/configuring-a-dmz-using-the-firewall-system-role.yml
     ```

     --ask-become-pass オプションは、**ansible-playbook** コマンドが **-u user_name** オプションで定義したユーザーの **sudo** パスワードを要求するようにします。

   -u user_name オプションを指定しないと、**ansible-playbook** は、コントロールノードに現在ログインしているユーザーとして管理ホストに接続します。

検証

- 管理ノードで、**dmz** ゾーンに関する詳細情報を表示します。

```
# firewall-cmd --zone=dmz --list-all
dmz (active)
  target: default
  icmp-block-inversion: no
  interfaces: enp1s0
  sources:
  services: https ssh
  ports:
  protocols:
  forward: no
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
```

1.17. 関連情報

- man ページの **firewalld(1)**
- **firewalld.conf(5)** の man ページ
- man ページの **firewall-cmd (1)**
- **firewall-config(1)** の man ページ
- **firewall-offline-cmd(1)** の man ページ
- **firewalld.icmptype(5)** の man ページ
- **firewalld.ipset(5)** の man ページ

- man ページの **firewalld.service(5)**
- man ページの **firewalld.zone(5)**
- **firewalld.direct(5)** の man ページ
- **firewalld.lockdown-whitelist(5)**
- **firewalld.richlanguage(5)**
- **firewalld.zones(5)** の man ページ
- **firewalld.dbus(5)** の man ページ

第2章 NFTABLES の使用

nftables フレームワークは、パケットの分類機能を提供します。最も重要な機能は次のとおりです。

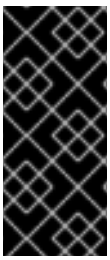
- 線形処理の代わりに組み込みルックアップテーブルを使用
- **IPv4** プロトコルおよび **IPv6** プロトコルに対する 1 つのフレームワーク
- 完全ルールセットのフェッチ、更新、および保存を行わず、すべてアトミックに適用されるルール
- ルールセットにおけるデバッグおよびトレースへの対応 (**nftrace**) およびトレースイベントの監視 (**nft** ツール)
- より統一されたコンパクトな構文、プロトコル固有の拡張なし
- サードパーティーのアプリケーション用 Netlink API

nftables フレームワークは、テーブルを使用してチェーンを保存します。このチェーンには、アクションを実行する個々のルールが含まれます。**libnftnl** ライブラリーは、**libmnl** ライブラリーの Netlink API の **nftables** で、低レベルの対話のために使用できます。

ルールセット変更が適用されていることを表示するには、**nft list ruleset** コマンドを使用します。これらのツールは、テーブル、チェーン、ルール、セットなどのオブジェクトを **nftables** ルールセットに追加するため、**nft flush ruleset** コマンドなどの **nftables** ルールセット操作は、先に別の従来のコマンドを使用してインストールしたルールセットに影響を及ぼす可能性があることに注意してください。

2.1. IPTABLES から NFTABLES への移行

ファイアウォール設定が依然として **iptables** ルールを使用している場合は、**iptables** ルールを **nftables** に移行できます。



重要

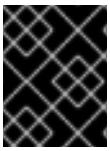
ipset パッケージおよび **iptables-nft** パッケージは、Red Hat Enterprise Linux 9で非推奨になりました。これには、**iptables**、**ip6tables**、**arptables**、および **ebtables** ユーティリティーなどの **nft-variants** の非推奨化が含まれます。以前のバージョンの RHEL からアップグレードしたなど、このツールのいずれかを使用している場合は、Red Hat は、**nftables** パッケージが提供する **nft** コマンドラインツールへの移行を推奨します。

2.1.1. firewalld、nftables、または iptables を使用する場合

以下は、次のユーティリティーのいずれかを使用する必要があるシナリオの概要です。

- **firewalld**:簡単なファイアウォールのユースケースには、**firewalld** ユーティリティーを使用します。このユーティリティーは、使いやすく、このようなシナリオの一般的な使用例に対応しています。
- **nftables**:**nftables** ユーティリティーを使用して、ネットワーク全体など、複雑なパフォーマンスに関する重要なファイアウォールを設定します。
- **iptables**:Red Hat Enterprise Linux の **iptables** ユーティリティーは、**legacy** バックエンドの代わりに **nf_tables** カーネル API を使用します。**nf_tables** API は、**iptables** コマンドを使用するスクリプトが、Red Hat Enterprise Linux で引き続き動作するように、後方互換性を提供しま

す。新しいファイアウォールスクリプトの場合には、Red Hat は **nftables** を使用することを推奨します。



重要

異なるファイアウォールサービスが相互に影響することを回避するには、RHEL ホストでそのうちの 1 つだけを実行し、他のサービスを無効にします。

2.1.2. iptables および ip6tables ルールセットの nftables への変換

iptables-restore-translate ユーティリティーおよび **ip6tables-restore-translate** ユーティリティーを使用して、**iptables** および **ip6tables** ルールセットを **nftables** に変換します。

前提条件

- **nftables** パッケージおよび **iptables** パッケージがインストールされている。
- システムに **iptables** ルールおよび **ip6tables** ルールが設定されている。

手順

1. **iptables** ルールおよび **ip6tables** ルールをファイルに書き込みます。

```
# iptables-save >/root/iptables.dump
# ip6tables-save >/root/ip6tables.dump
```

2. ダンプファイルを **nftables** 命令に変換します。

```
# iptables-restore-translate -f /root/iptables.dump > /etc/nftables/ruleset-migrated-from-iptables.nft
# ip6tables-restore-translate -f /root/ip6tables.dump > /etc/nftables/ruleset-migrated-from-ip6tables.nft
```

3. 必要に応じて、生成された **nftables** ルールを手動で更新して、確認します。
4. **nftables** サービスが生成されたファイルをロードできるようにするには、以下を **/etc/sysconfig/nftables.conf** ファイルに追加します。

```
include "/etc/nftables/ruleset-migrated-from-iptables.nft"
include "/etc/nftables/ruleset-migrated-from-ip6tables.nft"
```

5. **iptables** サービスを停止し、無効にします。

```
# systemctl disable --now iptables
```

カスタムスクリプトを使用して **iptables** ルールを読み込みた場合は、スクリプトが自動的に起動しなくなり、すべてのテーブルをフラッシュするために再起動するようにしてください。

6. **nftables** サービスを有効にして起動します。

```
# systemctl enable --now nftables
```

検証

- **nftables** ルールセットを表示します。

```
# nft list ruleset
```

関連情報

- システムの起動時に nftables ルールの自動読み込み

2.1.3. 単一の iptables ルールおよび ip6tables ルールを nftables に変換する

Red Hat Enterprise Linux は、**iptables** ルールまたは **ip6tables** ルールを、**nftables** で同等のルールに変換する **iptables-translate** ユーティリティーおよび **ip6tables-translate** ユーティリティーを提供します。

前提条件

- **nftables** パッケージがインストールされている。

手順

- **iptables** または **ip6tables** の代わりに **iptables-translate** ユーティリティーまたは **ip6tables-translate** ユーティリティーを使用して、対応する **nftables** ルールを表示します。以下に例を示します。

```
# iptables-translate -A INPUT -s 192.0.2.0/24 -j ACCEPT
nft add rule ip filter INPUT ip saddr 192.0.2.0/24 counter accept
```

拡張機能によっては変換機能がない場合もあります。このような場合には、ユーティリティーは、以下のように、前に **#** 記号が付いた未変換ルールを出力します。

```
# iptables-translate -A INPUT -j CHECKSUM --checksum-fill
nft # -A INPUT -j CHECKSUM --checksum-fill
```

関連情報

- **iptables-translate --help**

2.1.4. 一般的な iptables コマンドと nftables コマンドの比較

以下は、一般的な **iptables** コマンドと **nftables** コマンドの比較です。

- すべてのルールを一覧表示します。

iptables	nftables
iptables-save	**nft list ruleset**

- 特定のテーブルおよびチェーンを一覧表示します。

iptables	nftables
iptables -L	**nft list table ip filter**
iptables -L INPUT	**nft list chain ip filter INPUT**
iptables -t nat -L PREROUTING	**nft list chain ip nat PREROUTING**

nft コマンドは、テーブルおよびチェーンを事前に作成しません。これらは、ユーザーが手動で作成した場合にのみ存在します。

例:firewalld が生成するルールの一覧表示

```
# nft list table inet firewalld
# nft list table ip firewalld
# nft list table ip6 firewalld
```

2.1.5. 関連情報

- iptables:2 つのバリアントと nftables の関係

2.2. NFTABLES スクリプトの作成および実行

nftables フレームワークは、シェルスクリプトを使用してファイアウォールルールを維持するための主な利点を提供するネイティブのスクリプト環境を提供します。スクリプトの実行はアトミックです。つまり、システムがスクリプト全体を適用するか、エラーが発生した場合には実行を阻止することを意味します。これにより、ファイアウォールは常に一貫した状態になります。

さらに、管理者は、**nftables** スクリプト環境で以下を行うことができます。

- コメントの追加
- 変数の定義
- 他のルールセットファイルの組み込み

本セクションでは、この機能を使用する方法と、**nftables** スクリプトの作成方法と実行方法を説明します。

nftables パッケージをインストールすると、Red Hat Enterprise Linux が自動的に ***.nft** スクリプトを **/etc/nftables/** ディレクトリーに作成します。このスクリプトは、さまざまな目的でテーブルと空のチェーンを作成するコマンドが含まれます。

2.2.1. 対応している nftables スクリプトの形式

nftables スクリプト環境は、次の形式でスクリプトに対応します。

- **nft list ruleset** コマンドが、ルールセットを表示するのと同じ形式でスクリプトを作成できます。

```
#!/usr/sbin/nft -f
```

```
# Flush the rule set
flush ruleset

table inet example_table {
  chain example_chain {
    # Chain for incoming packets that drops all packets that
    # are not explicitly allowed by any rule in this chain
    type filter hook input priority 0; policy drop;

    # Accept connections to port 22 (ssh)
    tcp dport ssh accept
  }
}
```

- **nft** コマンドと同じ構文を使用できます。

```
#!/usr/sbin/nft -f

# Flush the rule set
flush ruleset

# Create a table
add table inet example_table

# Create a chain for incoming packets that drops all packets
# that are not explicitly allowed by any rule in this chain
add chain inet example_table example_chain { type filter hook input priority 0 ; policy drop ; }

# Add a rule that accepts connections to port 22 (ssh)
add rule inet example_table example_chain tcp dport ssh accept
```

2.2.2. nftables スクリプトの実行

nftables スクリプトを実行するには、**nft** ユーティリティーに渡すか、スクリプトを直接実行します。

前提条件

- このセクションの手順では、**nftables** スクリプトを **/etc/nftables/example_firewall.nft** ファイルに保存していることを前提としています。

手順

- **nftables** スクリプトを **nft** ユーティリティーに渡して実行するには、次のコマンドを実行します。

```
# nft -f /etc/nftables/example_firewall.nft
```

- **nftables** スクリプトを直接実行するには、次のコマンドを実行します。

  a. 以下の手順は、一度だけ必要です。

     i. スクリプトが以下のシバンシーケンスで始まることを確認します。

```
#!/usr/sbin/nft -f
```

重要

-f パラメーターを省略すると、**nft** ユーティリティーはスクリプトを読み込まず、**Error: syntax error, unexpected newline, expecting string** のように表示されます。

ii. オプション:スクリプトの所有者を **root** に設定します。

```
# chown root /etc/nftables/example_firewall.nft
```

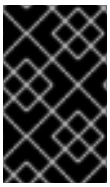
iii. 所有者のスクリプトを実行ファイルに変更します。

```
# chmod u+x /etc/nftables/example_firewall.nft
```

b. スクリプトを実行します。

```
# /etc/nftables/example_firewall.nft
```

出力が表示されない場合は、システムがスクリプトを正常に実行します。



重要

nft はスクリプトを正常に実行しますが、ルールの配置やパラメーター不足、またはスクリプト内のその他の問題により、ファイアウォールが期待通りの動作を起こさない可能性があります。

関連情報

- **chown(1)** の man ページ
- **chmod(1)** の man ページ
- システムの起動時に nftables ルールの自動読み込み

2.2.3. nftables スクリプトでコメントの使用

nftables スクリプト環境は、**#** 文字の右側にあるすべてをコメントとして解釈します。

例2.1 nftables スクリプトのコメント

コメントは、コマンドの横だけでなく、行の先頭からも開始できます。

```
...
# Flush the rule set
flush ruleset

add table inet example_table  # Create a table
...
```

2.2.4. nftables スクリプトで変数の使用

nftables スクリプトで変数を定義するには、**define** キーワードを使用します。シングル値および匿名セットを変数に保存できます。より複雑なシナリオの場合は、セットまたは決定マップを使用します。

値を 1 つ持つ変数

以下の例は、値が **enp1s0** の **INET_DEV** という名前の変数を定義します。

```
define INET_DEV = enp1s0
```

スクリプトで変数を使用するには、**$** 記号と、それに続く変数名を指定します。

```
...
add rule inet example_table example_chain iifname $INET_DEV tcp dport ssh accept
...
```

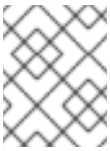
匿名セットを含む変数

以下の例では、匿名セットを含む変数を定義します。

```
define DNS_SERVERS = { 192.0.2.1, 192.0.2.2 }
```

スクリプトで変数を使用するには、**$** 記号と、それに続く変数名を指定します。

```
add rule inet example_table example_chain ip daddr $DNS_SERVERS accept
```



注記

中括弧は、変数がセットを表していることを示すため、ルールで使用する場合は、特別なセマンティクスを持つことに注意してください。

関連情報

- nftables コマンドでのセットの使用
- nftables コマンドにおける決定マップの使用

2.2.5. nftables スクリプトへのファイルの追加

nftables スクリプト環境を使用すると、管理者は **include** ステートメントを使用して他のスクリプトを追加できます。

絶対パスまたは相対パスのないファイル名のみを指定すると、**nftables** には、デフォルトの検索パスのファイルが含まれます。これは、Red Hat Enterprise Linux では **/etc** に設定されています。

例2.2 デフォルト検索ディレクトリーからのファイルを含む

デフォルトの検索ディレクトリーからファイルを指定するには、次のコマンドを実行します。

```
include "example.nft"
```

例2.3 ディレクトリーの *.nft ファイルをすべて含む

***.nft** で終わるすべてのファイルを **/etc/nftables/rulesets/** ディレクトリーに保存するには、次のコマンドを実行します。

```
include "/etc/nftables/rulesets/*.nft"
```

include ステートメントは、ドットで始まるファイルに一致しないことに注意してください。

関連情報

- **nft(8)** のmanページの **Include files** セクション

2.2.6. システムの起動時に nftables ルールの自動読み込み

systemd サービス **nftables** は、**/etc/sysconfig/nftables.conf** ファイルに含まれるファイアウォールスクリプトを読み込みます。本セクションでは、システムの起動時にファイアウォールルールを読み込む方法を説明します。

前提条件

- **nftables** スクリプトは、**/etc/nftables/** ディレクトリーに保存されます。

手順

1. **/etc/sysconfig/nftables.conf** ファイルを編集します。
   - **nftables** パッケージをインストールしたときに **/etc/nftables/** に作成された ***.nft** スクリプトを拡張する場合は、そのスクリプトの **include** ステートメントのコメントを解除します。
   - スクリプトを新規に作成する場合は、そのスクリプトを含む **include** ステートメントを追加します。たとえば、**nftables** サービスの起動時に **/etc/nftables/example.nft** スクリプトを読み込むには、以下を追加します。

     ```
     include "/etc/nftables/example.nft"
     ```

2. 必要に応じて、**nftables** サービスを開始して、システムを再起動せずにファイアウォールルールを読み込みます。

   ```
   # systemctl start nftables
   ```

3. **nftables** サービスを有効にします。

   ```
   # systemctl enable nftables
   ```

関連情報

- 対応している nftables スクリプトの形式

2.3. NFTABLES テーブル、チェーン、およびルールの作成および管理

本セクションでは、**nftables** ルールセットを表示する方法と、その管理方法を説明します。

2.3.1. 標準のチェーン優先度の値およびテキスト名

チェーンを作成する場合は、**priority** で、同じ **hook** 値を持つチェーンが通過する順番を指定する、整数値または標準名を設定できます。

名前と値は、デフォルトのチェーンの登録時に **xtables** が使用する優先度に基づいて定義されます。

注記

nft list chains コマンドは、デフォルトで文字列の優先順位の値を表示します。**-y** オプションをコマンドに渡すことで、数値を表示できます。

例2.4 テキスト値を使用した優先順位の設定

以下のコマンドは、標準の優先度 **50** を使用して **example_table** に **example_chain** という名前のチェーンを作成します。

```
# nft add chain inet example_table example_chain { type filter hook input priority 50 \; policy accept \; }
```

優先度は標準の値であるため、テキスト値を使用することもできます。

```
# nft add chain inet example_table example_chain { type filter hook input priority security \; policy accept \; }
```

表2.1 標準優先順位名、ファミリー、およびフックの互換性マトリックス

名前	値	ファミリー	フック
raw	-300	**ip**、**ip6**、**inet**	all
mangle	-150	**ip**、**ip6**、**inet**	all
dstnat	-100	**ip**、**ip6**、**inet**	prerouting
filter	0	**ip**、**ip6**、**inet**、**arp**、**netdev**	all
security	50	**ip**、**ip6**、**inet**	all
srcnat	100	**ip**、**ip6**、**inet**	postrouting

すべてのファミリーは同じ値を使用しますが、**bridge** ファミリーは以下の値を使用します。

表2.2 ブリッジファミリーの標準的な優先順位名およびフックの互換性

名前	値	フック
dstnat	-300	prerouting

名前	値	フック
filter	-200	all
out	100	出力
srcnat	300	postrouting

関連情報

- **nft(8)** のmanページの **Chains** セクション

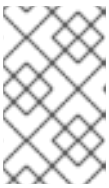
2.3.2. nftables ルールセットの表示

nftables のルールセットには、テーブル、チェーン、およびルールが含まれます。本セクションでは、ルールセットを表示する方法を説明します。

手順

- ルールセットを表示するには、以下のコマンドを実行します。

```
# nft list ruleset
table inet example_table {
  chain example_chain {
    type filter hook input priority filter; policy accept;
    tcp dport http accept
    tcp dport ssh accept
  }
}
```



注記

デフォルトで、**nftables** は事前作成のテーブルではありません。したがって、テーブルがないホストに設定したルールを表示すると、**nft list ruleset** コマンドが出力を表示しません。

2.3.3. nftables テーブルの作成

nftables の表は、チェーン、ルール、セットなどのオブジェクトを含む名前空間です。本セクションでは、テーブルの作成方法を説明します。

各テーブルには、アドレスファミリーが定義されている必要があります。テーブルのアドレスファミリーは、テーブルプロセスのアドレスタイプを定義します。テーブルを作成する際に、以下のいずれかのアドレスファミリーを設定できます。

- **ip**:IPv4 パケットだけに一致します。アドレスファミリーを指定しないと、これがデフォルトになります。
- **ip6**:IPv6 パケットだけに一致します。

- **inet**:IPv4 パケットと IPv6 パケットの両方に一致します。
- **arp**:IPv4 アドレス解決プロトコル (ARP) パケットに一致します。
- **bridge**:ブリッジデバイスを通過するパケットに一致します。
- **netdev**:ingress からのパケットに一致します。

手順

1. **nft add table** コマンドを使用してテーブルを新規作成します。たとえば、IPv4 パケットおよび IPv6 パケットを処理する **example_table** という名前のテーブルを作成するには、次のコマンドを実行します。

```
# nft add table inet example_table
```

2. 必要に応じて、ルールセットのテーブルを一覧表示します。

```
# nft list tables
table inet example_table
```

関連情報

- **nft(8)** のmanページの **Address families** セクション
- **nft(8)** のmanページの **Tables** セクション

2.3.4. nftables チェーンの作成

チェーンは、ルールのコンテナーです。次の 2 つのルールタイプが存在します。

- ベースチェーン:ネットワークスタックからのパケットのエントリーポイントとしてベースチェーンを使用できます。
- 通常のチェーン:**ジャンプ** ターゲットとして通常のチェーンを使用し、ルールをより適切に整理できます。

この手順では、既存のテーブルにベースチェーンを追加する方法を説明します。

前提条件

- 新しいチェーンを追加するテーブルが存在する。

手順

1. **nft add chain** コマンドを使用してチェーンを新規作成します。たとえば、**example_table** に、**example_chain** という名前のチェーンを作成するには、次のコマンドを実行します。

```
# nft add chain inet example_table example_chain { type filter hook input priority 0 \; policy accept \; }
```

重要

シェルでセミコロンがコマンドの最後として解釈されるのを回避するには、セミコロンの前に \ エスケープ文字を付けます。

このチェーンは、着信パケットをフィルタリングします。**priority** パラメーターは、**nftables** プロセスが同じフック値を持つチェーンを処理する順序を指定します。優先度の値が低いほど優先されます。**policy** パラメーターは、このチェーン内のルールのデフォルトアクションを設定します。サーバーにリモートでログインし、デフォルトのポリシーを **drop** に設定すると、他のルールでリモートアクセスが許可されていなければ、すぐに切断されることに注意してください。

2. 必要に応じて、すべてのチェーンを表示します。

```
# nft list chains
table inet example_table {
  chain example_chain {
    type filter hook input priority filter; policy accept;
  }
}
```

関連情報

- **nft(8)** のmanページの **Address families** セクション
- **nft(8)** のmanページの **Chains** セクション

2.3.5. nftables チェーンの最後に対するルールの追加

本セクションでは、既存の **nftables** チェーンの最後にルールを追加する方法を説明します。

前提条件

- ルールを追加するチェーンが存在する。

手順

1. 新規ルールを追加するには、**nft add rule** コマンドを使用します。たとえば、**example_table** の **example_chain** に、ポート 22 の TCP トラフィックを許可するルールを追加するには、次のコマンドを実行します。

```
# nft add rule inet example_table example_chain tcp dport 22 accept
```

ポート番号の代わりに、サービス名を指定することもできます。この例では、ポート番号 **22** の代わりに **ssh** を使用できます。サービス名は、**/etc/services** ファイルのエントリーに基づいてポート番号に解決されることに注意してください。

2. 必要に応じて、**example_table** ですべてのチェーンとそのルールを表示します。

```
# nft list table inet example_table
table inet example_table {
  chain example_chain {
    type filter hook input priority filter; policy accept;
    ...
```

```
    tcp dport ssh accept
  }
}
```

関連情報

- **nft(8)** のmanページの **Address families** セクション
- **nft(8)** のmanページの **Rules** セクション

2.3.6. nftables チェーンの先頭へのルールの挿入

本セクションでは、既存の **nftables** チェーンの先頭にルールを追加する方法を説明します。

前提条件

- ルールを追加するチェーンが存在する。

手順

1. 新しいルールを追加するには、**nft insert rule** コマンドを使用します。たとえば、ポート 22 で TCP トラフィックを許可するルールを **example_table** の **example_chain** に追加するには、次のコマンドを実行します。

```
# nft insert rule inet example_table example_chain tcp dport 22 accept
```

代わりに、ポート番号の代わりにサービスの名前を指定することもできます。この例では、ポート番号 **22** の代わりに **ssh** を使用できます。サービス名は、**/etc/services** ファイルのエントリーに基づいてポート番号に解決されることに注意してください。

2. 必要に応じて、**example_table** ですべてのチェーンとそのルールを表示します。

```
# nft list table inet example_table
table inet example_table {
  chain example_chain {
    type filter hook input priority filter; policy accept;
    tcp dport ssh accept
    ...
  }
}
```

関連情報

- **nft(8)** のmanページの **Address families** セクション
- **nft(8)** のmanページの **Rules** セクション

2.3.7. nftables チェーンの特定の位置へのルールの挿入

本セクションでは、**nftables** チェーンで、既存のルールの前後にルールを追加する方法を説明します。これにより、正しい場所に新しいルールを配置することができます。

前提条件

- ルールを追加するチェーンが存在する。

手順

1. **nft -a list ruleset** コマンドを使用して、ハンドルなど、**example_table** 内のすべてのチェーンとそのルールを表示します。

```
# nft -a list table inet example_table
table inet example_table { # handle 1
  chain example_chain { # handle 1
    type filter hook input priority filter; policy accept;
    tcp dport 22 accept # handle 2
    tcp dport 443 accept # handle 3
    tcp dport 389 accept # handle 4
  }
}
```

   -a を使用すると、ハンドルが表示されます。次の手順で新しいルールを配置するときに、この情報が必要です。

2. **example_table** の **example_chain** チェーンに新しいルールを挿入します。

   - ハンドル **3** の前に、ポート **636** で TCP トラフィックを許可するルールを挿入するには、次のコマンドを実行します。

```
# nft insert rule inet example_table example_chain position 3 tcp dport 636 accept
```

   - ハンドル **3** の後ろに、ポート **80** で TCP トラフィックを許可するルールを追加するには、次のコマンドを実行します。

```
# nft add rule inet example_table example_chain position 3 tcp dport 80 accept
```

3. 必要に応じて、**example_table** ですべてのチェーンとそのルールを表示します。

```
# nft -a list table inet example_table
table inet example_table { # handle 1
  chain example_chain { # handle 1
    type filter hook input priority filter; policy accept;
    tcp dport 22 accept # handle 2
    tcp dport 636 accept # handle 5
    tcp dport 443 accept # handle 3
    tcp dport 80 accept # handle 6
    tcp dport 389 accept # handle 4
  }
}
```

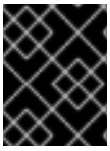
関連情報

- **nft(8)** のmanページの **Address families** セクション
- **nft(8)** のmanページの **Rules** セクション

2.4. NFTABLES を使用した NAT の設定

nftables を使用すると、以下のネットワークアドレス変換 (NAT) タイプを設定できます。

- マスカレーディング
- ソース NAT (SNAT)
- 宛先 NAT (DNAT)
- リダイレクト



重要

iifname パラメーターおよび **oifname** パラメーターでは実インターフェース名のみを使用でき、代替名 (**altname**) には対応していません。

2.4.1. 異なる NAT タイプ: マスカレード、ソース NAT、宛先 NAT、リダイレクト

以下は、さまざまなネットワークアドレス変換 (NAT) タイプになります。

マスカレードおよびソースの NAT (SNAT)

この NAT タイプのいずれかを使用して、パケットのソース IP アドレスを変更します。たとえば、インターネットサービスプロバイダーは、プライベート IP 範囲 (**10.0.0.0/8** など) をルーティングしません。ネットワークでプライベート IP 範囲を使用し、ユーザーがインターネット上のサーバーにアクセスできるようにする必要がある場合は、この範囲のパケットのソース IP アドレスをパブリック IP アドレスにマップします。
マスカレードおよび SNAT の両方は非常に似ています。相違点は次のとおりです。

- マスカレードは、出力インターフェースの IP アドレスを自動的に使用します。したがって、出力インターフェースが動的 IP アドレスを使用する場合は、マスカレードを使用します。
- SNAT は、パケットのソース IP アドレスを指定された IP に設定し、出力インターフェースの IP アドレスを動的に検索しません。そのため、SNAT の方がマスカレードよりも高速です。出力インターフェースが固定 IP アドレスを使用する場合は、SNAT を使用します。

宛先 NAT (DNAT)

この NAT タイプを使用して、着信パケットの宛先アドレスとポートを書き換えます。たとえば、Web サーバーがプライベート IP 範囲の IP アドレスを使用しているため、インターネットから直接アクセスできない場合は、ルーターに DNAT ルールを設定し、着信トラフィックをこのサーバーにリダイレクトできます。

リダイレクト

このタイプは、チェーンフックに応じてパケットをローカルマシンにリダイレクトする DNAT の特殊なケースです。たとえば、サービスが標準ポートとは異なるポートで実行する場合は、標準ポートからこの特定のポートに着信トラフィックをリダイレクトすることができます。

2.4.2. nftables を使用したマスカレードの設定

マスカレードを使用すると、ルーターは、インターフェースを介して送信されるパケットのソース IP を、インターフェースの IP アドレスに動的に変更できます。これは、インターフェースに新しい IP が割り当てられている場合に、**nftables** はソース IP の置き換え時に新しい IP を自動的に使用することを意味します。

次の手順では、**ens3** インターフェイスを介して、ホストから **ens3** の IP セットに送信されるパケットのソース IP を置き換える方法を説明します。

手順

1. テーブルを作成します。

```
# nft add table nat
```

2. テーブルに、**prerouting** チェーンおよび **postrouting** チェーンを追加します。

```
# nft -- add chain nat prerouting { type nat hook prerouting priority -100 \; }
# nft add chain nat postrouting { type nat hook postrouting priority 100 \; }
```

重要

prerouting チェーンにルールを追加しなくても、**nftables** フレームワークでは、着信パケット返信に一致するようにこのチェーンが必要になります。

nft コマンドに **--** オプションを渡すと、シェルが優先度の負の値を **nft** コマンドのオプションとして解釈する必要がなくなります。

3. **postrouting** チェーンに、**ens3** インターフェースの出力パケットに一致するルールを追加します。

```
# nft add rule nat postrouting oifname "ens3" masquerade
```

2.4.3. nftables を使用したソース NAT の設定

ルーターでは、ソース NAT (SNAT) を使用して、インターフェースを介して特定の IP アドレスに送信するパケットの IP を変更できます。

次の手順では、**ens3** インターフェイスを介して、ルーターから **192.0.2.1** に送信されるパケットのソース IP を置き換える方法を説明します。

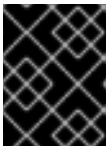
手順

1. テーブルを作成します。

```
# nft add table nat
```

2. テーブルに、**prerouting** チェーンおよび **postrouting** チェーンを追加します。

```
# nft -- add chain nat prerouting { type nat hook prerouting priority -100 \; }
# nft add chain nat postrouting { type nat hook postrouting priority 100 \; }
```



重要

postrouting チェーンにルールを追加しなくても、**nftables** フレームワークでは、このチェーンが発信パケット返信に一致するようにする必要があります。

nft コマンドに **--** オプションを渡すと、シェルが優先度の負の値を **nft** コマンドのオプションとして解釈する必要がなくなります。

3. **ens3** を介した発信パケットのソース IP を **192.0.2.1** に置き換えるルールを **postrouting** チェーンに追加します。

```
# nft add rule nat postrouting oifname "ens3" snat to 192.0.2.1
```