Red Hat Enterprise Linux 8

Redes de segurança

Configuração de redes seguras e comunicação em rede

Last Updated: 2021-03-05

Red Hat Enterprise Linux 8 Redes de segurança

Configuração de redes seguras e comunicação em rede

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.

Nota Legal

Resumo

Este título auxilia os administradores a proteger redes, máquinas conectadas e comunicação em rede contra vários ataques.

Índice

TORNANDO O CÓDIGO ABERTO MAIS INCLUSIVO

A Red Hat tem o compromisso de substituir a linguagem problemática em nosso código, documentação e propriedades da web. Estamos começando com estes quatro termos: master, slave, blacklist e whitelist. Por causa da enormidade deste esforço, estas mudanças serão implementadas gradualmente ao longo de vários lançamentos futuros. Para mais detalhes, veja a mensagem de nosso CTO Chris Wright.

FORNECENDO FEEDBACK SOBRE A DOCUMENTAÇÃO DA RED HAT

Agradecemos sua contribuição em nossa documentação. Por favor, diga-nos como podemos melhorá-la. Para fazer isso:

- Para comentários simples sobre passagens específicas:
  1. Certifique-se de que você está visualizando a documentação no formato *Multi-page HTML*. Além disso, certifique-se de ver o botão **Feedback** no canto superior direito do documento.
  2. Use o cursor do mouse para destacar a parte do texto que você deseja comentar.
  3. Clique no pop-up **Add Feedback** que aparece abaixo do texto destacado.
  4. Siga as instruções apresentadas.
- Para enviar comentários mais complexos, crie um bilhete Bugzilla:
  1. Ir para o site da Bugzilla.
  2. Como Componente, use **Documentation**.
  3. Preencha o campo **Description** com sua sugestão de melhoria. Inclua um link para a(s) parte(s) relevante(s) da documentação.
  4. Clique em **Submit Bug**.

CAPÍTULO 1. USANDO COMUNICAÇÕES SEGURAS ENTRE DOIS SISTEMAS COM OPENSSH

SSH (Secure Shell) é um protocolo que fornece comunicações seguras entre dois sistemas usando uma arquitetura cliente-servidor e permite que os usuários façam login em sistemas host de servidores remotamente. Ao contrário de outros protocolos de comunicação remota, como FTP ou Telnet, o SSH criptografa a sessão de login, o que impede que intrusos coletem senhas não criptografadas da conexão.

O Red Hat Enterprise Linux inclui os pacotes básicos **OpenSSH**: o pacote geral **openssh**, o pacote **openssh-server** e o pacote **openssh-clients**. Note que os pacotes **OpenSSH** requerem o pacote **OpenSSL openssl-libs** , que instala várias bibliotecas criptográficas importantes que permitem que **OpenSSH** forneça comunicações criptografadas.

1.1. SSH E OPENSSH

SSH (Secure Shell) é um programa para efetuar login em uma máquina remota e executar comandos nessa máquina. O protocolo SSH fornece comunicações criptografadas seguras entre dois hosts não confiáveis através de uma rede insegura. Você também pode encaminhar conexões X11 e portas TCP/IP arbitrárias através do canal seguro.

O protocolo SSH atenua as ameaças à segurança, tais como interceptação da comunicação entre dois sistemas e imitação de um determinado host, quando você o utiliza para login remoto ou cópia de arquivo. Isto porque o cliente e o servidor SSH usam assinaturas digitais para verificar suas identidades. Além disso, toda a comunicação entre os sistemas cliente e servidor é criptografada.

OpenSSH é uma implementação do protocolo SSH suportada por uma série de sistemas operacionais Linux, UNIX e similares. Ele inclui os arquivos centrais necessários tanto para o cliente OpenSSH quanto para o servidor. A suíte OpenSSH consiste das seguintes ferramentas de espaço do usuário:

- **ssh** é um programa de login remoto (cliente SSH)
- **sshd** é um **OpenSSH** daemon SSH
- **scp** é um programa seguro de cópia remota de arquivos
- **sftp** é um programa seguro de transferência de arquivos
- **ssh-agent** é um agente de autenticação para o cache de chaves privadas
- **ssh-add** adiciona identidades chave privadas a **ssh-agent**
- **ssh-keygen** gera, gerencia e converte chaves de autenticação para **ssh**
- **ssh-copy-id** é um script que adiciona chaves públicas locais ao arquivo **authorized_keys** em um servidor SSH remoto
- **ssh-keyscan** - reúne as chaves de anfitrião público do SSH

Existem atualmente duas versões do SSH: a versão 1, e a mais recente versão 2. A suíte **OpenSSH** no Red Hat Enterprise Linux 8 suporta apenas o SSH versão 2, que tem um algoritmo melhorado de troca de chaves não vulnerável a explorações conhecidas na versão 1.

OpenSSH, como um dos subsistemas criptográficos centrais da RHEL, utiliza políticas de criptografia em todo o sistema. Isto assegura que os conjuntos de cifras fracas e algoritmos criptográficos sejam desativados na configuração padrão. Para ajustar a política, o administrador deve usar o comando

update-crypto-policies para fazer configurações mais rígidas ou mais frouxas ou optar manualmente pela exclusão das políticas criptográficas de todo o sistema.

A suíte **OpenSSH** utiliza dois conjuntos diferentes de arquivos de configuração: aqueles para programas de clientes (ou seja, **ssh**, **scp** e **sftp**), e aqueles para o servidor (o daemon **sshd**). As informações de configuração SSH de todo o sistema são armazenadas no diretório **/etc/ssh/**. As informações de configuração do SSH específicas do usuário são armazenadas em **~/.ssh/**, no diretório home do usuário. Para uma lista detalhada dos arquivos de configuração do OpenSSH, consulte a seção **FILES** na página de manual **sshd(8)**.

Recursos adicionais

- Páginas de homens para o tópico **ssh** listado pelo comando **man -k ssh**.
- Usando políticas criptográficas de todo o sistema .

1.2. CONFIGURANDO E INICIANDO UM SERVIDOR OPENSSH

Use o seguinte procedimento para uma configuração básica que possa ser necessária para seu ambiente e para iniciar um servidor **OpenSSH**. Observe que após a instalação padrão da RHEL, o daemon **sshd** já foi iniciado e as chaves do servidor são criadas automaticamente.

Pré-requisitos

- O pacote **openssh-server** está instalado.

Procedimento

1. Inicie o daemon **sshd** na sessão atual e configure-o para iniciar automaticamente no momento da inicialização:

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

2. Para especificar endereços diferentes do padrão **0.0.0.0** (IPv4) ou **::** (IPv6) para a diretiva **ListenAddress** no arquivo de configuração **/etc/ssh/sshd_config** e para usar uma configuração de rede dinâmica mais lenta, adicione a dependência da unidade alvo **network-online.target** ao arquivo de unidade **sshd.service**. Para conseguir isso, crie o arquivo **/etc/systemd/system/sshd.service.d/local.conf** com o seguinte conteúdo:

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

3. Analise se as configurações do servidor **OpenSSH** no arquivo de configuração **/etc/ssh/sshd_config** atendem aos requisitos de seu cenário.

4. Opcionalmente, altere a mensagem de boas-vindas que seu servidor **OpenSSH** exibe antes que um cliente se autentique, editando o arquivo **/etc/issue**, por exemplo:

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

Certifique-se de que a opção **Banner** não seja comentada em **/etc/ssh/sshd_config** e seu valor contenha **/etc/issue**:

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

Note que para alterar a mensagem exibida após um login bem sucedido, você tem que editar o arquivo **/etc/motd** no servidor. Consulte a página de manual **pam_motd** para maiores informações.

5. Recarregue a configuração **systemd** e reinicie **sshd** para aplicar as mudanças:

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

Etapas de verificação

1. Verifique se o daemon **sshd** está funcionando:

```
# systemctl status sshd
● sshd.service - OpenSSH server daemon
   Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2019-11-18 14:59:58 CET; 6min ago
     Docs: man:sshd(8)
           man:sshd_config(5)
 Main PID: 1149 (sshd)
    Tasks: 1 (limit: 11491)
   Memory: 1.9M
   CGroup: /system.slice/sshd.service
           └─1149 /usr/sbin/sshd -D -oCiphers=aes128-ctr,aes256-ctr,aes128-cbc,aes256-cbc -
oMACs=hmac-sha2-256,>

Nov 18 14:59:58 ssh-server-example.com systemd[1]: Starting OpenSSH server daemon...
Nov 18 14:59:58 ssh-server-example.com sshd[1149]: Server listening on 0.0.0.0 port 22.
Nov 18 14:59:58 ssh-server-example.com sshd[1149]: Server listening on :: port 22.
Nov 18 14:59:58 ssh-server-example.com systemd[1]: Started OpenSSH server daemon.
```

2. Conecte-se ao servidor SSH com um cliente SSH.

```
# ssh user@ssh-server-example.com
ECDSA key fingerprint is SHA256:dXbaS0RG/UzlTTku8GtXSz0S1++lPegSy31v3L/FAEc.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ssh-server-example.com' (ECDSA) to the list of known hosts.

user@ssh-server-example.com's password:
```

Recursos adicionais

- **sshd(8)** e **sshd_config(5)** páginas man

1.3. USANDO PARES DE CHAVES AO INVÉS DE SENHAS PARA AUTENTICAÇÃO SSH

Para melhorar ainda mais a segurança do sistema, gerar pares de chaves SSH e depois impor a autenticação baseada em chaves, desativando a autenticação por senha.

1.3.1. Configurando um servidor OpenSSH para autenticação baseada em chaves

Siga estes passos para configurar seu servidor OpenSSH para fazer cumprir a autenticação baseada em chaves.

Pré-requisitos

- O pacote **openssh-server** está instalado.
- O daemon **sshd** está rodando no servidor.

Procedimento

1. Abra a configuração **/etc/ssh/sshd_config** em um editor de texto, por exemplo:

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

2. Mude a opção **PasswordAuthentication** para **no**:

```
SenhaAutenticação não
```

   Em um sistema que não seja uma nova instalação padrão, verifique se **PubkeyAuthentication no** não foi definido e se a diretiva **ChallengeResponseAuthentication** está definida para **no**. Se você estiver conectado remotamente, não usando console ou acesso fora da banda, teste o processo de login baseado em chave antes de desativar a autenticação da senha.

3. Para utilizar a autenticação baseada em chaves com diretórios domésticos montados em NFS, habilite o **use_nfs_home_dirs** SELinux boolean:

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

4. Recarregue o daemon **sshd** para aplicar as mudanças:

```
# systemctl reload sshd
```

Recursos adicionais

- **sshd(8)**, **sshd_config(5)**, e **setsebool(8)** páginas man

1.3.2. Geração de pares de chaves SSH

Use este procedimento para gerar um par de chaves SSH em um sistema local e para copiar a chave pública gerada para um servidor **OpenSSH**. Se o servidor estiver configurado de acordo, você pode entrar no servidor **OpenSSH** sem fornecer nenhuma senha.

IMPORTANTE

Se você completar os seguintes passos como **root**, somente **root** é capaz de usar as chaves.

Procedimento

1. Para gerar um par de chaves ECDSA para a versão 2 do protocolo SSH:

```
$ ssh-keygen -t ecdsa
Generating public/private ecdsa key pair.
Enter file in which to save the key (/home/joesec/.ssh/id_ecdsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/joesec/.ssh/id_ecdsa.
Your public key has been saved in /home/joesec/.ssh/id_ecdsa.pub.
The key fingerprint is:
SHA256:Q/x+qms4j7PCQ0qFd09iZEFHA+SqwBKRNaU72oZfaCI
joesec@localhost.example.com
The key's randomart image is:
+---[ECDSA 256]---+
|.oo..o=++        |
|.. o .oo .       |
|. .. o. o        |
|....o.+...       |
|o.oo.o +S .      |
|.=.+.   .o       |
|E.*+.  .  . .    |
|.=..+ +..  o     |
|  .  oo*+o.      |
+----[SHA256]-----+
```

Você também pode gerar um par de chaves RSA usando a opção **-t rsa** com o comando **ssh-keygen** ou um par de chaves Ed25519, digitando o comando **ssh-keygen -t ed25519**.

2. Para copiar a chave pública para uma máquina remota:

```
$ ssh-copy-id joesec@ssh-server-example.com
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
joesec@ssh-server-example.com's password:
...
Number of key(s) added: 1

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

Se você não usar o programa **ssh-agent** em sua sessão, o comando anterior copia a chave pública ~/**.ssh**/**id*.pub** modificada mais recentemente, caso ainda não esteja instalada. Para especificar outro arquivo de chave pública ou para priorizar chaves em arquivos sobre chaves armazenadas em cache na memória por **ssh-agent**, use o comando **ssh-copy-id** com a opção **-i**.

NOTA

Se você reinstalar seu sistema e quiser manter os pares de chaves gerados anteriormente, faça backup do diretório ~/**.ssh**/. Após a reinstalação, copie-o de volta para seu diretório home. Você pode fazer isso para todos os usuários em seu sistema, incluindo **root**.

Etapas de verificação

1. Acesse o servidor OpenSSH sem fornecer nenhuma senha:

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

Recursos adicionais

- **ssh-keygen(1)** e **ssh-copy-id(1)** páginas man

1.4. USANDO CHAVES SSH ARMAZENADAS EM UM CARTÃO INTELIGENTE

O Red Hat Enterprise Linux 8 permite que você use chaves RSA e ECDSA armazenadas em um cartão inteligente em clientes OpenSSH. Use este procedimento para habilitar a autenticação usando um Cartão Smart Card ao invés de usar uma senha.

Pré-requisitos

- No lado do cliente, o pacote **opensc** está instalado e o serviço **pcscd** está funcionando.

Procedimento

1. Liste todas as chaves fornecidas pelo módulo OpenSC PKCS #11 incluindo seus PKCS #11 URIs e salve a saída para o arquivo *keys.pub*:

```
$ ssh-keygen -D pkcs11: > keys.pub
$ ssh-keygen -D pkcs11:
ssh-rsa AAAAB3NzaC1yc2E...KKZMzcQZzx
pkcs11:id=%02;object=SIGN%20pubkey;token=SSH%20key;manufacturer=piv_II?module-
path=/usr/lib64/pkcs11/opensc-pkcs11.so
ecdsa-sha2-nistp256 AAA...J0hkYnnsM=
pkcs11:id=%01;object=PIV%20AUTH%20pubkey;token=SSH%20key;manufacturer=piv_II?
module-path=/usr/lib64/pkcs11/opensc-pkcs11.so
```

2. Para permitir a autenticação usando um cartão inteligente em um servidor remoto (*example.com*), transfira a chave pública para o servidor remoto. Use o comando **ssh-copy-id** com *keys.pub* criado na etapa anterior:

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

3. Para conectar-se a *example.com* usando a chave ECDSA da saída do comando **ssh-keygen -D** no passo 1, você pode usar apenas um subconjunto do URI, que faz referência única à sua chave, por exemplo:

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

4. Você pode usar a mesma cadeia URI no arquivo ~/**.ssh**/**config** para tornar a configuração permanente:

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

Como o OpenSSH usa a embalagem **p11-kit-proxy** e o módulo OpenSC PKCS #11 está registrado no Kit PKCS #11, você pode simplificar os comandos anteriores:

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

Se você pular a parte **id=** de um PKCS #11 URI, o OpenSSH carrega todas as chaves que estão disponíveis no módulo proxy. Isto pode reduzir a quantidade de digitação necessária:

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

Recursos adicionais

- Fedora 28: Melhor suporte a cartões inteligentes no OpenSSH
- **p11-kit(8)** página do homem
- **ssh(1)** página do homem
- **ssh-keygen(1)** página do homem
- **opensc.conf(5)** página do homem
- **pcscd(8)** página do homem

1.5. TORNANDO O OPENSSH MAIS SEGURO

As seguintes dicas o ajudam a aumentar a segurança ao usar o OpenSSH. Note que as mudanças no arquivo de configuração /**etc**/**ssh**/**sshd_config** OpenSSH requerem o recarregamento do daemon **sshd** para ter efeito:

```
# systemctl reload sshd
```

IMPORTANTE

A maioria das mudanças de configuração de endurecimento de segurança reduz a compatibilidade com clientes que não suportam algoritmos atualizados ou conjuntos de cifras.

Desabilitando protocolos de conexão inseguros

- Para tornar o SSH verdadeiramente eficaz, evite o uso de protocolos de conexão inseguros que são substituídos pela suíte **OpenSSH**. Caso contrário, a senha de um usuário pode ser protegida usando SSH para apenas uma sessão, a ser capturada posteriormente ao fazer o login usando Telnet. Por este motivo, considere desativar protocolos inseguros, tais como telnet, rsh, rlogin e ftp.

Habilitação da autenticação baseada em chave e desativação da autenticação baseada em senha

- Desabilitar senhas para autenticação e permitir apenas pares de chaves reduz a superfície de ataque e também pode poupar o tempo dos usuários. Em clientes, gerar pares de chaves usando a ferramenta **ssh-keygen** e usar o utilitário **ssh-copy-id** para copiar chaves públicas de clientes no servidor **OpenSSH**. Para desativar a autenticação baseada em senhas em seu servidor OpenSSH, edite **/etc/ssh/sshd_config** e mude a opção **PasswordAuthentication** para **no**:

```
SenhaAutenticação não
```

Tipos de chaves

- Embora o comando **ssh-keygen** gere um par de chaves RSA por padrão, você pode instruí-lo a gerar chaves ECDSA ou Ed25519 usando a opção **-t**. O ECDSA (Elliptic Curve Digital Signature Algorithm) oferece melhor desempenho do que o RSA com a força simétrica equivalente da chave. Ele também gera chaves mais curtas. O algoritmo de chave pública Ed25519 é uma implementação de curvas Edwards retorcidas que é mais segura e também mais rápida que RSA, DSA, e ECDSA.
  O OpenSSH cria automaticamente chaves de servidor RSA, ECDSA e Ed25519 se elas estiverem faltando. Para configurar a criação da chave de host no RHEL 8, use o serviço instanciado **sshd-keygen@.service**. Por exemplo, para desativar a criação automática do tipo de chave RSA:

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

- Para excluir tipos-chave específicos para conexões SSH, comente as linhas relevantes em **/etc/ssh/sshd_config**, e recarregue o serviço **sshd**. Por exemplo, para permitir apenas as chaves de host Ed25519:

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

Porto sem falta

- Por padrão, o daemon **sshd** ouve na porta TCP 22. A mudança da porta reduz a exposição do sistema a ataques baseados em varredura automatizada da rede e, assim, aumenta a segurança através da obscuridade. Você pode especificar a porta usando a diretiva **Port** no arquivo de configuração **/etc/ssh/sshd_config**.
  Você também tem que atualizar a política padrão da SELinux para permitir o uso de uma porta não inadimplente. Para fazer isso, utilize a ferramenta **semanage** do pacote **policycoreutils-python-utils**:

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

  Além disso, atualizar a configuração **firewalld**:

```
# firewall-cmd --add-port port_number/tcp
# firewall-cmd --runtime-to-permanent
```

Nos comandos anteriores, substituir *port_number* pelo novo número de porta especificado usando a diretiva **Port**.

Sem login de raiz

- Se seu caso particular de uso não exigir a possibilidade de login como usuário root, você deve considerar a possibilidade de configurar a diretiva de configuração **PermitRootLogin** para **no** no arquivo **/etc/ssh/sshd_config**. Desativando a possibilidade de logar-se como usuário root, o administrador pode auditar quais usuários executam que comandos privilegiados depois de logar-se como usuários regulares e depois ganhar direitos de root.
  Alternativamente, defina **PermitRootLogin** para **prohibit-password**:

  ```
  PermitRootLogin proíbe palavras-passe
  ```

  Isto reforça o uso de autenticação baseada em chaves em vez do uso de senhas para o login como raiz e reduz os riscos ao prevenir ataques de força bruta.

Usando a extensão X Security

- O servidor X em clientes Red Hat Enterprise Linux não fornece a extensão X Security. Portanto, os clientes não podem solicitar outra camada de segurança ao conectar-se a servidores SSH não confiáveis com o encaminhamento X11. A maioria das aplicações não é capaz de rodar com esta extensão habilitada de qualquer forma.
  Por padrão, a opção **ForwardX11Trusted** no arquivo **/etc/ssh/ssh_config.d/05-redhat.conf** está definida para **yes**, e não há diferença entre o comando **ssh -X remote_machine** (host não confiável) e **ssh -Y remote_machine** (trust host).

  Se seu cenário não exigir o recurso de encaminhamento X11, defina a diretiva **X11Forwarding** no arquivo de configuração **/etc/ssh/sshd_config** para **no**.

Restringir o acesso a usuários, grupos ou domínios específicos

- As diretrizes **AllowUsers** e **AllowGroups** no servidor de arquivos de configuração **/etc/ssh/sshd_config** permitem que você permita que somente determinados usuários, domínios ou grupos se conectem ao seu servidor OpenSSH. Você pode combinar **AllowUsers** e **AllowGroups** para restringir o acesso de forma mais precisa, por exemplo:

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

  As linhas de configuração anteriores aceitam conexões de todos os usuários dos sistemas em 192.168.1.* e 10.0.0.* sub-redes, exceto do sistema com o endereço 192.168.1.2. Todos os usuários devem estar no grupo **example-group**. O servidor OpenSSH nega todas as outras conexões.

  Observe que o uso de listas de permissão (diretrizes que começam com Allow) é mais seguro do que o uso de listas de bloco (opções que começam com Deny) porque as listas de permissão também bloqueiam novos usuários ou grupos não autorizados.

Mudando as políticas criptográficas de todo o sistema

- **OpenSSH** utiliza as políticas criptográficas do sistema RHEL, e o nível padrão de políticas criptográficas do sistema oferece configurações seguras para os modelos de ameaça atuais. Para tornar suas configurações criptográficas mais rígidas, altere o nível da política atual:

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

- Para optar pela exclusão das políticas de criptografia de todo o sistema para seu servidor **OpenSSH**, descomente a linha com a variável **CRYPTO_POLICY=** no arquivo **/etc/sysconfig/sshd**. Após esta mudança, os valores que você especificar nas seções **Ciphers**, **MACs**, **KexAlgoritms**, e **GSSAPIKexAlgorithms** no arquivo **/etc/ssh/sshd_config** não serão sobrepostos. Note que esta tarefa requer profunda experiência na configuração de opções criptográficas.

- Veja Utilizando políticas criptográficas em todo o sistema no título de endurecimento de segurança RHEL 8 para mais informações.

Recursos adicionais

- **sshd_config(5)**, **ssh-keygen(1)**, **crypto-policies(7)**, e **update-crypto-policies(8)** páginas man

1.6. CONEXÃO A UM SERVIDOR REMOTO USANDO UM HOST SSH JUMP

Use este procedimento para conectar-se a um servidor remoto através de um servidor intermediário, também chamado de jump host.

Pré-requisitos

- Um host de salto aceita conexões SSH de seu sistema.
- Um servidor remoto aceita conexões SSH somente a partir do host de salto.

Procedimento

1. Defina o host de salto, editando o arquivo **~/.ssh/config**, por exemplo:

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

2. Adicione a configuração de salto do servidor remoto com a diretiva **ProxyJump** para **~/.ssh/config**, por exemplo:

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

3. Conecte-se ao servidor remoto através do servidor de salto:

```
$ ssh remote-server
```

O comando anterior é equivalente ao comando **ssh -J jump-server1 remote-server** se você omitir os passos de configuração 1 e 2.

NOTA

Você pode especificar mais servidores de salto e também pode pular a adição de definições de host ao arquivo de configurações quando você fornecer seus nomes de host completos, por exemplo:

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

Mude a notação do nome do host apenas no comando anterior se os nomes dos usuários ou portas SSH nos servidores de salto forem diferentes dos nomes e portas no servidor remoto, por exemplo:

```
$ ssh -J
johndoe@jump1.example.com:75,johndoe@jump2.example.com:75,johndoe@jump3.e
xample.com:75 joesec@remote1.example.com:220
```

Recursos adicionais

- **ssh_config(5)** e **ssh(1)** páginas man

1.7. CONEXÃO A MÁQUINAS REMOTAS COM CHAVES SSH USANDO O SSH-AGENT

Para evitar a entrada de uma senha cada vez que você inicia uma conexão SSH, você pode usar o utilitário **ssh-agent** para fazer o cache da chave SSH privada. A chave privada e a frase-senha permanecem seguras.

Pré-requisitos

- Você tem um host remoto com daemon SSH rodando e alcançável através da rede.
- Você sabe o endereço IP ou o nome do host e as credenciais para fazer o login no host remoto.
- Você gerou um par de chaves SSH com uma frase-chave e transferiu a chave pública para a máquina remota. Para mais informações, consulte Gerando pares de chaves SSH .

Procedimento

1. Opcional: Verifique se você pode usar a chave para autenticar no host remoto:
   a. Conecte-se ao host remoto usando SSH:

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

   b. Digite a frase-chave que você definiu ao criar a chave para conceder acesso à chave privada.

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

2. Inicie o **ssh-agent**.

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

3. Adicione a chave a **ssh-agent**.

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

Etapas de verificação

- Opcional: Faça o login na máquina host usando SSH.

```
$ ssh example.user1@198.51.100.1

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

Note que você não precisou digitar a senha.

1.8. RECURSOS ADICIONAIS

Para mais informações sobre configuração e conexão com servidores e clientes do **OpenSSH** no Red Hat Enterprise Linux, veja os recursos listados abaixo.

Documentação instalada

- **sshd(8)** página man documenta as opções de linha de comando disponíveis e fornece uma lista completa de arquivos de configuração e diretórios suportados.
- a página de manual**ssh(1)** fornece uma lista completa de opções de linha de comando disponíveis e arquivos de configuração e diretórios suportados.
- a página de manual**scp(1)** fornece uma descrição mais detalhada da utilidade **scp** e seu uso.
- a página de manual**sftp(1)** fornece uma descrição mais detalhada da utilidade **sftp** e seu uso.
- **ssh-keygen(1)** man page documents in detail the use of the **ssh-keygen** utility to generate, manage, and convert authentication keys used by ssh.
- a página de manual**ssh-copy-id(1)** descreve o uso do roteiro **ssh-copy-id**.
- **ssh_config(5)** página man documentos disponíveis opções de configuração do cliente SSH.
- a página de manual**sshd_config(5)** fornece uma descrição completa das opções de configuração do daemon SSH disponíveis.
- a página de manual**update-crypto-policies(8)** fornece orientação sobre o gerenciamento de políticas criptográficas de todo o sistema
- **crypto-policies(7)** página man fornece uma visão geral dos níveis de política criptográfica de todo o sistema

Documentação on-line

- OpenSSH Home Page - contém mais documentação, perguntas freqüentes, links para as listas de discussão, relatórios de bugs e outros recursos úteis.
- Configurando o SELinux para aplicações e serviços com configurações não-padrão - você pode aplicar procedimentos análogos para OpenSSH em uma configuração não-padrão com SELinux em modo de aplicação.
- Controle do tráfego da rede usando firewalld - fornece orientação sobre a atualização das configurações **firewalld** após a mudança de uma porta SSH

CAPÍTULO 2. PLANEJAMENTO E IMPLEMENTAÇÃO DO TLS

O TLS (Transport Layer Security) é um protocolo criptográfico usado para proteger as comunicações em rede. Ao endurecer as configurações de segurança do sistema através da configuração de protocolos preferenciais de troca de chaves, métodos de autenticação e algoritmos de criptografia, é necessário ter em mente que quanto mais ampla for a gama de clientes suportados, menor será a segurança resultante. Por outro lado, configurações rígidas de segurança levam a uma compatibilidade limitada com os clientes, o que pode resultar no bloqueio de alguns usuários fora do sistema. Certifique-se de visar a configuração mais estrita disponível e só a relaxe quando for necessária por razões de compatibilidade.

2.1. PROTOCOLOS SSL E TLS

O protocolo Secure Sockets Layer (SSL) foi originalmente desenvolvido pela Netscape Corporation para fornecer um mecanismo para comunicação segura através da Internet. Posteriormente, o protocolo foi adotado pela Internet Engineering Task Force (IETF) e renomeado para Transport Layer Security (TLS).

O protocolo TLS fica entre uma camada de protocolo de aplicação e uma camada de transporte confiável, como o TCP/IP. Ele é independente do protocolo de aplicação e, portanto, pode ser estratificado sob muitos protocolos diferentes, por exemplo: HTTP, FTP, SMTP, e assim por diante.

Versão do protocolo	Recomendação de uso
SSL v2	Não usar. Tem sérias vulnerabilidades de segurança. Removido das bibliotecas criptográficas do núcleo desde a RHEL 7.
SSL v3	Não usar. Tem sérias vulnerabilidades de segurança. Removido das bibliotecas criptográficas do núcleo desde a RHEL 8.
TLS 1.0	Não é recomendado o uso. Tem problemas conhecidos que não podem ser mitigados de forma a garantir a interoperabilidade, e não suporta as modernas suítes de cifras. Ativado somente no perfil de política criptográfica do sistema **LEGACY**.
TLS 1.1	Utilizar para fins de interoperabilidade onde for necessário. Não suporta os modernos conjuntos de cifras. Ativado somente na política **LEGACY**.
TLS 1.2	Suporta as modernas suítes de cifras AEAD. Esta versão está habilitada em todas as políticas de criptografia do sistema, mas as partes opcionais deste protocolo contêm vulnerabilidades e o TLS 1.2 também permite algoritmos desatualizados.
TLS 1.3	Versão recomendada. O TLS 1.3 remove opções problemáticas conhecidas, proporciona privacidade adicional ao encriptar mais do aperto de mão da negociação e pode ser mais rápido graças ao uso de algoritmos criptográficos modernos mais eficientes. O TLS 1.3 também é habilitado em todas as políticas de criptografia de todo o sistema.

Recursos adicionais

- IETF: O Protocolo de Segurança da Camada de Transporte (TLS) Versão 1.3

2.2. CONSIDERAÇÕES DE SEGURANÇA PARA TLS EM RHEL 8

No RHEL 8, as considerações relacionadas à criptografia são significativamente simplificadas graças às políticas de criptografia de todo o sistema. A política de criptografia **DEFAULT** permite apenas o TLS 1.2 e 1.3. Para permitir que seu sistema negocie conexões usando as versões anteriores do TLS, você precisa optar por não seguir as políticas de criptografia em uma aplicação ou mudar para a política **LEGACY** com o comando **update-crypto-policies**. Consulte Utilizando políticas criptográficas de todo o sistema para obter mais informações.

As configurações padrão fornecidas pelas bibliotecas incluídas no RHEL 8 são suficientemente seguras para a maioria das implantações. As implementações do TLS utilizam algoritmos seguros sempre que possível, sem impedir conexões de ou para clientes ou servidores legados. Aplique configurações reforçadas em ambientes com requisitos rígidos de segurança onde clientes ou servidores legados que não suportam algoritmos ou protocolos seguros não são esperados ou permitidos para conexão.

A maneira mais simples de endurecer sua configuração de TLS é mudar o nível de política criptográfica do sistema para **FUTURE** usando o comando **update-crypto-policies --set FUTURE**.

Se você decidir não seguir as políticas de criptografia do sistema RHEL, use as seguintes recomendações para protocolos preferenciais, conjuntos de cifras e comprimentos de chave em sua configuração personalizada:

2.2.1. Protocolos

A última versão do TLS fornece o melhor mecanismo de segurança. A menos que você tenha uma razão convincente para incluir suporte para versões antigas do TLS, permita que seus sistemas negociem conexões usando pelo menos a versão 1.2 do TLS. Observe que apesar de a RHEL 8 suportar a versão 1.3 do TLS, nem todas as características deste protocolo são totalmente suportadas pelos componentes da RHEL 8. Por exemplo, a característica 0-RTT (Zero Round Trip Time), que reduz a latência da conexão, ainda não é totalmente suportada pelos servidores web Apache ou Nginx.

2.2.2. Suítes de cifras

As suítes de cifras modernas e mais seguras devem ser preferidas às antigas e inseguras. Sempre desabilite o uso das suítes de cifras eNULL e aNULL, que não oferecem nenhuma criptografia ou autenticação. Se possível, as suítes de cifras baseadas em RC4 ou HMAC-MD5, que têm sérias deficiências, também devem ser desativadas. O mesmo se aplica às chamadas suítes de cifras de exportação, que foram intencionalmente tornadas mais fracas e, portanto, são fáceis de quebrar.

Embora não imediatamente inseguras, as suítes de cifras que oferecem menos de 128 bits de segurança não devem ser consideradas para sua curta vida útil. Algoritmos que usam 128 bits de segurança ou mais podem ser inquebráveis por pelo menos vários anos e, portanto, são fortemente recomendados. Note que enquanto as cifras 3DES anunciam o uso de 168 bits, elas realmente oferecem 112 bits de segurança.

Sempre dê preferência a conjuntos de cifras que suportem (perfeito) sigilo (PFS), o que garante a confidencialidade dos dados criptografados, mesmo no caso da chave do servidor ser comprometida. Isto exclui a rápida troca de chaves RSA, mas permite o uso de ECDHE e DHE. Dos dois, o ECDHE é o mais rápido e, portanto, a escolha preferida.

Você também deve dar preferência às cifras AEAD, como AES-GCM, antes das cifras CBC-mode, pois elas não são vulneráveis a ataques de oráculos de enchimento. Além disso, em muitos casos, o AES-GCM é mais rápido que o AES em modo CBC, especialmente quando o hardware possui aceleradores criptográficos para AES.

Observe também que ao utilizar a troca de chaves ECDHE com certificados ECDSA, a transação é ainda

mais rápida do que a troca de chaves RSA puras. Para dar suporte aos clientes antigos, é possível instalar dois pares de certificados e chaves em um servidor: um com chaves ECDSA (para novos clientes) e outro com chaves RSA (para os clientes antigos).

2.2.3. Comprimento da chave pública

Ao usar chaves RSA, sempre prefira chaves com pelo menos 3072 bits assinadas por pelo menos SHA-256, que é suficientemente grande para 128 bits de segurança verdadeiros.

ATENÇÃO

A segurança de seu sistema é apenas tão forte quanto o elo mais fraco da cadeia. Por exemplo, uma cifra forte por si só não garante uma boa segurança. As chaves e os certificados são tão importantes quanto as funções de hash e as chaves usadas pela Autoridade de Certificação (AC) para assinar suas chaves.

Recursos adicionais

- Políticas de criptografia de todo o sistema no RHEL 8 .
- **update-crypto-policies(8)** página do homem

2.3. CONFIGURAÇÃO DE TLS ENDURECIMENTO EM APLICAÇÕES

No Red Hat Enterprise Linux 8, as políticas de criptografia de todo o sistema fornecem uma maneira conveniente de garantir que suas aplicações usando bibliotecas criptográficas não permitam protocolos, cifras ou algoritmos inseguros conhecidos.

Se você quiser endurecer sua configuração relacionada ao TLS com suas configurações criptográficas personalizadas, você pode usar as opções de configuração criptográfica descritas nesta seção, e substituir as políticas de criptografia de todo o sistema apenas na quantidade mínima necessária.

Independentemente da configuração que você escolher usar, certifique-se sempre de exigir que seu aplicativo de servidor faça cumprir *server-side cipher order* , de modo que o conjunto de cifras a ser usado seja determinado pela ordem que você configurar.

2.3.1. Configurando o Apache HTTP server

O **Apache HTTP Server** pode usar tanto as bibliotecas **OpenSSL** como **NSS** para suas necessidades de TLS. O Red Hat Enterprise Linux 8 fornece a funcionalidade **mod_ssl** através de pacotes epônimos:

```
# yum instalar mod_ssl
```

O pacote **mod_ssl** instala o arquivo de configuração **/etc/httpd/conf.d/ssl.conf**, que pode ser usado para modificar as configurações relacionadas ao TLS do **Apache HTTP Server**.

Instale o pacote **httpd-manual** para obter a documentação completa para o **Apache HTTP Server**, incluindo a configuração do TLS. As diretrizes disponíveis no arquivo de configuração **/etc/httpd/conf.d/ssl.conf** são descritas em detalhes em /usr/share/httpd/manual/mod/mod_ssl.html.

Exemplos de várias configurações estão em /usr/share/httpd/manual/ssl/ssl_howto.html.

Ao modificar as configurações no arquivo de configuração **/etc/httpd/conf.d/ssl.conf**, não deixe de considerar as três diretrizes a seguir no mínimo:

SSLProtocol

Use esta diretiva para especificar a versão do TLS ou SSL que você deseja permitir.

SSLCipherSuite

Use esta diretiva para especificar seu conjunto de cifras preferido ou desabilite aqueles que você deseja desautorizar.

SSLHonorCipherOrder

Descomente e defina esta diretiva para **on** para garantir que os clientes de conexão adiram à ordem de cifras que você especificou.

Por exemplo, utilizar somente o protocolo TLS 1.2 e 1.3:

```
SSLProtocol             all -SSLv3 -TLSv1 -TLSv1.1
```

2.3.2. Configurando o servidor Nginx HTTP e proxy

Para ativar o suporte ao TLS 1.3 em **Nginx**, adicione o valor **TLSv1.3** à opção **ssl_protocols** na seção **server** do arquivo de configuração **/etc/nginx/nginx.conf**:

```
server {
    listen 443 ssl http2;
    listen [::]:443 ssl http2;
    ....
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers
    ....
}
```

2.3.3. Configuração do servidor de correio do Dovecot

Para configurar sua instalação do servidor de e-mail **Dovecot** para usar o TLS, modifique o arquivo de configuração **/etc/dovecot/conf.d/10-ssl.conf**. Você pode encontrar uma explicação de algumas das diretrizes básicas de configuração disponíveis nesse arquivo no arquivo /usr/share/doc/dovecot/wiki/SSL.DovecotConfiguration.txt, que é instalado junto com a instalação padrão de **Dovecot**.

Ao modificar as configurações no arquivo de configuração **/etc/dovecot/conf.d/10-ssl.conf**, não deixe de considerar as três diretrizes a seguir no mínimo:

ssl_protocols

Use esta diretiva para especificar a versão do TLS ou SSL que você deseja permitir ou desativar.

ssl_cipher_list

Use esta diretiva para especificar suas suítes de cifras preferidas ou desativar as que você deseja desautorizar.

ssl_prefer_server_ciphers

Descomente e defina esta diretiva para **yes** para garantir que os clientes de conexão adiram à ordem de cifras que você especificou.

Por exemplo, a seguinte linha em **/etc/dovecot/conf.d/10-ssl.conf** permite apenas o TLS 1.1 e posteriores:

```
ssl_protocols = !SSLv2 !SSLv3 !TLSv1
```

Recursos adicionais

Para mais informações sobre a configuração do TLS e tópicos relacionados, veja os recursos listados abaixo.

- a página de manual**config(5)** descreve o formato do arquivo de configuração **/etc/ssl/openssl.conf**.
- **ciphers(1)** página man inclui uma lista de palavras-chave e cifras disponíveis **OpenSSL**.
- Recomendações para o uso seguro da camada de segurança de transporte (TLS) e da camada de segurança de transporte de datagramas (DTLS)
- OMozilla SSL Configuration Generator pode ajudar a criar arquivos de configuração para **Apache** ou **Nginx** com configurações seguras que desabilitam protocolos vulneráveis conhecidos, cifras e algoritmos de hashing.
- Oteste do servidor SSL verifica se sua configuração atende aos modernos requisitos de segurança.

CAPÍTULO 3. CONFIGURAÇÃO DE UMA VPN COM IPSEC

No Red Hat Enterprise Linux 8, uma rede privada virtual (VPN) pode ser configurada usando o protocolo **IPsec**, que é suportado pela aplicação **Libreswan**.

3.1. LIBRESWAN COMO UMA IMPLEMENTAÇÃO DA VPN IPSEC

No Red Hat Enterprise Linux 8, uma Rede Privada Virtual (VPN) pode ser configurada usando o protocolo **IPsec**, que é suportado pela aplicação **Libreswan**. **Libreswan** é uma continuação da aplicação **Openswan**, e muitos exemplos da documentação **Openswan** são intercambiáveis com **Libreswan**.

O protocolo **IPsec** para uma VPN é configurado usando o protocolo Internet Key Exchange (**IKE**). Os termos IPsec e IKE são utilizados de forma intercambiável. Uma VPN IPsec também é chamada de IKE VPN, IKEv2 VPN, XAUTH VPN, Cisco VPN ou IKE/IPsec VPN. Uma variante de uma VPN IPsec que também usa o Protocolo de Tunelamento de Nível 2 (**L2TP**) é normalmente chamada de VPN L2TP/IPsec, que requer a aplicação do canal opcional **xl2tpd**.

Libreswan é uma implementação open-source, espaço do usuário **IKE**. **IKE** v1 e v2 são implementados como um daemon em nível de usuário. O protocolo IKE também é criptografado. O protocolo **IPsec** é implementado pelo kernel Linux, e **Libreswan** configura o kernel para adicionar e remover configurações de túneis VPN.

O protocolo **IKE** utiliza as portas UDP 500 e 4500. O protocolo **IPsec** consiste em dois protocolos:

- Encapsulated Security Payload (**ESP**), que tem o protocolo número 50.
- Cabeçalho Autenticado (**AH**), que tem o protocolo número 51.

O protocolo **AH** não é recomendado para uso. Recomenda-se aos usuários do **AH** que migrem para **ESP** com criptografia nula.

O protocolo **IPsec** oferece dois modos de operação:

- **Tunnel Mode** (o padrão)
- **Transport Mode**.

Você pode configurar o kernel com IPsec sem o IKE. Isto é chamado **Manual Keying**. Você também pode configurar a digitação manual usando os comandos **ip xfrm**, no entanto, isto é fortemente desencorajado por razões de segurança. **Libreswan** faz interface com o kernel Linux usando o netlink. A criptografia e descriptografia de pacotes acontecem no kernel Linux.

Libreswan utiliza a biblioteca criptográfica Network Security Services (**NSS**). Tanto **Libreswan** quanto **NSS** estão certificados para uso com a *Federal Information Processing Standard* (**FIPS**) Publicação 140-2.

IMPORTANTE

IKE/**IPsec** VPNs, implementadas por **Libreswan** e pelo kernel Linux, é a única tecnologia VPN recomendada para uso no Red Hat Enterprise Linux 8. Não utilize nenhuma outra tecnologia VPN sem compreender os riscos de fazê-lo.

No Red Hat Enterprise Linux 8, **Libreswan** segue **system-wide cryptographic policies** por default. Isto assegura que **Libreswan** usa configurações seguras para os modelos de ameaça atuais, incluindo **IKEv2**

como um protocolo default. Veja Usando políticas de criptografia em todo o sistema para mais informações.

Libreswan não usa os termos "fonte" e "destino" ou "servidor" e "cliente" porque IKE/IPsec são protocolos peer to peer. Em vez disso, usa os termos "esquerda" e "direita" para se referir aos pontos finais (os anfitriões). Isto também permite utilizar a mesma configuração em ambos os pontos finais na maioria dos casos. Entretanto, os administradores geralmente optam por usar sempre "esquerda" para o host local e "direita" para o host remoto.

3.2. INSTALAÇÃO DE LIBRESWAN

Este procedimento descreve os passos para instalar e iniciar a implementação da VPN **Libreswan** IPsec/IKE.

Pré-requisitos

- O repositório **AppStream** está habilitado.

Procedimento

1. Instale os pacotes **libreswan**:

```
# yum install libreswan
```

2. Se você estiver reinstalando **Libreswan**, remova seus arquivos antigos de banco de dados:

```
# systemctl stop ipsec
# rm /etc/ipsec.d/*db
```

3. Iniciar o serviço **ipsec**, e permitir que o serviço seja iniciado automaticamente na inicialização:

```
# systemctl enable ipsec --now
```

4. Configure o firewall para permitir portas 500 e 4500/UDP para os protocolos IKE, ESP, e AH adicionando o serviço **ipsec**:

```
# firewall-cmd --add-service="ipsec"
# firewall-cmd --runtime-to-permanent
```

3.3. CRIANDO UMA VPN HOSPEDEIRO-A-ANFITRIÃO

Para configurar **Libreswan** para criar uma VPN de host a host **IPsec** entre dois hosts referidos como *left* e *right*, digite os seguintes comandos em ambos os hosts:

Procedimento

1. Gerar um par de chaves RSA em cada host:

```
# ipsec newhostkey --output /etc/ipsec.d/hostkey.secrets
```

2. A etapa anterior retornou a chave gerada **ckaid**. Use essa **ckaid** com o seguinte comando em *left*, por exemplo:

```
# ipsec showhostkey --left --ckaid 2d3ea57b61c9419dfd6cf43a1eb6cb306c0e857d
```

A saída do comando anterior gerou a linha **leftrsasigkey=** necessária para a configuração. Faça o mesmo no segundo host (*right*):

```
# ipsec showhostkey --right --ckaid a9e1f6ce9ecd3608c24e8f701318383f41798f03
```

3. No diretório **/etc/ipsec.d/**, crie um novo arquivo **my_host-to-host.conf**. Escreva as chaves do host RSA a partir da saída dos comandos **ipsec showhostkey** no passo anterior para o novo arquivo. Por exemplo:

```
conn mytunnel
    leftid=@west
    left=192.1.2.23
    leftrsasigkey=0sAQOrlo+hOafUZDlCQmXFrje/oZm [...] W2n417C/4urYHQkCvuIQ==
    rightid=@east
    right=192.1.2.45
    rightrsasigkey=0sAQO3fwC6nSSGgt64DWiYZzuHbc4 [...] D/v8t5YTQ==
    authby=rsasig
```

4. Após a importação das chaves, reinicie o serviço **ipsec**:

```
# systemctl restart ipsec
```

5. Iniciar **Libreswan**:

```
# ipsec setup start
```

6. Carregue a conexão:

```
# ipsec auto --add mytunnel
```

7. Estabelecer o túnel:

```
# ipsec auto --up mytunnel
```

8. Para iniciar automaticamente o túnel quando o serviço **ipsec** for iniciado, adicione a seguinte linha à definição da conexão:

```
auto=arranque
```

3.4. CONFIGURAÇÃO DE UMA VPN SITE A SITE

Para criar um site a site **IPsec** VPN, unindo duas redes, é criado um túnel **IPsec** entre os dois hosts. Os hosts atuam assim como pontos finais, que são configurados para permitir a passagem do tráfego de uma ou mais sub-redes. Portanto, pode-se pensar no host como gateways para a parte remota da rede.

A configuração da VPN site-a-site só difere da VPN host-a-host na medida em que uma ou mais redes ou sub-redes devem ser especificadas no arquivo de configuração.

Pré-requisitos

- Uma VPN hospedeiro-a-anfitrião já está configurada.

Procedimento

1. Copie o arquivo com a configuração de sua VPN de host a host para um novo arquivo, por exemplo:

```
# cp /etc/ipsec.d/my_host-to-host.conf /etc/ipsec.d/my_site-to-site.conf
```

2. Adicione a configuração da sub-rede ao arquivo criado na etapa anterior, por exemplo:

```
conn mysubnet
    also=mytunnel
    leftsubnet=192.0.1.0/24
    rightsubnet=192.0.2.0/24
    auto=start

conn mysubnet6
    also=mytunnel
    leftsubnet=2001:db8:0:1::/64
    rightsubnet=2001:db8:0:2::/64
    auto=start

# the following part of the configuration file is the same for both host-to-host and site-to-site
connections:

conn mytunnel
   leftid=@west
   left=192.1.2.23
   leftrsasigkey=0sAQOrlo+hOafUZDlCQmXFrje/oZm [...] W2n417C/4urYHQkCvuIQ==
   rightid=@east
   right=192.1.2.45
   rightrsasigkey=0sAQO3fwC6nSSGgt64DWiYZzuHbc4 [...] D/v8t5YTQ==
   authby=rsasig
```

3.5. CONFIGURAÇÃO DE UMA VPN DE ACESSO REMOTO

Guerreiros de estrada são usuários que viajam com clientes móveis com um endereço IP atribuído dinamicamente, como os laptops. Os clientes móveis autenticam usando certificados.

O exemplo a seguir mostra a configuração para **IKEv2**, e evita o uso do protocolo **IKEv1** XAUTH.

No servidor:

```
conn roadwarriors
   ikev2=insist
   # Support (roaming) MOBIKE clients (RFC 4555)
   mobike=yes
   fragmentation=yes
   left=1.2.3.4
   # if access to the LAN is given, enable this, otherwise use 0.0.0.0/0
   # leftsubnet=10.10.0.0/16
   leftsubnet=0.0.0.0/0
   leftcert=gw.example.com
```

```
    leftid=%fromcert
    leftxauthserver=yes
    leftmodecfgserver=yes
    right=%any
    # trust our own Certificate Agency
    rightca=%same
    # pick an IP address pool to assign to remote users
    # 100.64.0.0/16 prevents RFC1918 clashes when remote users are behind NAT
    rightaddresspool=100.64.13.100-100.64.13.254
    # if you want remote clients to use some local DNS zones and servers
    modecfgdns="1.2.3.4, 5.6.7.8"
    modecfgdomains="internal.company.com, corp"
    rightxauthclient=yes
    rightmodecfgclient=yes
    authby=rsasig
    # optionally, run the client X.509 ID through pam to allow/deny client
    # pam-authorize=yes
    # load connection, don't initiate
    auto=add
    # kill vanished roadwarriors
    dpddelay=1m
    dpdtimeout=5m
    dpdaction=clear
```

No cliente móvel, o dispositivo do guerreiro da estrada, utiliza uma pequena variação da configuração anterior:

```
conn to-vpn-server
    ikev2=insist
    # pick up our dynamic IP
    left=%defaultroute
    leftsubnet=0.0.0.0/0
    leftcert=myname.example.com
    leftid=%fromcert
    leftmodecfgclient=yes
    # right can also be a DNS hostname
    right=1.2.3.4
    # if access to the remote LAN is required, enable this, otherwise use 0.0.0.0/0
    # rightsubnet=10.10.0.0/16
    rightsubnet=0.0.0.0/0
    fragmentation=yes
    # trust our own Certificate Agency
    rightca=%same
    authby=rsasig
    # allow narrowing to the server’s suggested assigned IP and remote subnet
    narrowing=yes
    # Support (roaming) MOBIKE clients (RFC 4555)
    mobike=yes
    # Initiate connection
    auto=start
```

3.6. CONFIGURAÇÃO DE UMA VPN EM MALHA

Uma rede mesh VPN, também conhecida como *any-to-any* VPN, é uma rede onde todos os nós se comunicam usando **IPsec**. A configuração permite exceções para os nós que não podem utilizar **IPsec**. A rede mesh VPN pode ser configurada de duas maneiras:

- Para requerer **IPsec**.
- Preferir **IPsec**, mas permitir um retorno para a comunicação de texto claro.

A autenticação entre os nós pode ser baseada em certificados X.509 ou em extensões de segurança DNS (DNSSEC).

O procedimento a seguir utiliza certificados X.509. Estes certificados podem ser gerados usando qualquer tipo de sistema de gerenciamento da Autoridade Certificadora (CA), como o Sistema de Certificado Dogtag. A Dogtag assume que os certificados para cada nó estão disponíveis no formato PKCS #12 (arquivos .p12), que contém a chave privada, o certificado do nó e o certificado Root CA usado para validar os certificados X.509 dos outros nós.

Cada nó tem uma configuração idêntica, com exceção de seu certificado X.509. Isto permite adicionar novos nós sem reconfigurar nenhum dos nós existentes na rede. Os arquivos PKCS #12 requerem um "nome amigável", para o qual usamos o nome "nó" para que os arquivos de configuração referentes ao nome amigável possam ser idênticos para todos os nós.

Pré-requisitos

- **Libreswan** está instalado, e o serviço **ipsec** é iniciado em cada nó.

Procedimento

1. Em cada nó, importar arquivos PKCS #12. Esta etapa requer a senha utilizada para gerar os arquivos PKCS #12:

```
# ipsec import nodeXXX.p12
```

2. Criar as três seguintes definições de conexão para os perfis **IPsec required** (privado), **IPsec optional** (privado ou claro), e **No IPsec** (claro):

```
# cat /etc/ipsec.d/mesh.conf
conn clear
 auto=ondemand
 type=passthrough
 authby=never
 left=%defaultroute
 right=%group

conn private
 auto=ondemand
 type=transport
 authby=rsasig
 failureshunt=drop
 negotiationshunt=drop
 # left
 left=%defaultroute
 leftcert=nodeXXXX
 leftid=%fromcert
       leftrsasigkey=%cert
 # right
```

```
 rightrsasigkey=%cert
 rightid=%fromcert
 right=%opportunisticgroup

conn private-or-clear
 auto=ondemand
 type=transport
 authby=rsasig
 failureshunt=passthrough
 negotiationshunt=passthrough
 # left
 left=%defaultroute
 leftcert=nodeXXXX
 leftid=%fromcert
       leftrsasigkey=%cert
 # right
 rightrsasigkey=%cert
 rightid=%fromcert
 right=%opportunisticgroup
```

3. Adicionar o endereço IP da rede na categoria apropriada. Por exemplo, se todos os nós residem na rede 10.15.0.0/16, e todos os nós devem ordenar a criptografia **IPsec**:

```
# echo "10.15.0.0/16" >> /etc/ipsec.d/policies/private
```

4. Para permitir que certos nós, por exemplo, 10.15.34.0/24, trabalhem com e sem **IPsec**, adicione esses nós ao grupo privado ou claro utilizando:

```
# echo "10.15.34.0/24" >> /etc/ipsec.d/policies/private-or-clear
```

5. Para definir um anfitrião, por exemplo, 10.15.1.2, que não seja capaz de **IPsec** no grupo claro, use:

```
# echo "10.15.1.2/32" >> /etc/ipsec.d/policies/clear
```

   Os arquivos no diretório **/etc/ipsec.d/policies** podem ser criados a partir de um modelo para cada novo nó, ou podem ser provisionados usando o Puppet ou o Ansible.

   Observe que cada nó tem a mesma lista de exceções ou diferentes expectativas de fluxo de tráfego. Dois nós, portanto, podem não ser capazes de se comunicar porque um requer **IPsec** e o outro não pode usar **IPsec**.

6. Reinicie o nó para adicioná-lo à malha configurada:

```
# systemctl restart ipsec
```

7. Uma vez terminada a adição de nós, um comando **ping** é suficiente para abrir um túnel **IPsec**. Para ver quais túneis um nó abriu:

```
# ipsec trafficstatus
```

3.7. MÉTODOS DE AUTENTICAÇÃO UTILIZADOS EM LIBRESWAN

Você pode usar os seguintes métodos para autenticação dos pontos finais:

- *Pre-Shared Keys* (PSK) é o método de autenticação mais simples. As PSKs devem consistir de caracteres aleatórios e ter um comprimento de pelo menos 20 caracteres. No modo FIPS, as PSKs precisam obedecer a um requisito de força mínima, dependendo do algoritmo de integridade utilizado. Recomenda-se não usar PSKs com menos de 64 caracteres aleatórios.

- *Raw RSA keys* são comumente usados para configurações estáticas de host-to-host ou subrede-a-subrede **IPsec**. Os hosts são configurados manualmente com a chave pública RSA um do outro. Este método não é bem dimensionado quando dezenas ou mais hosts precisam configurar túneis **IPsec** uns para os outros.

- *X.509 certificates* são comumente usados para implantações em larga escala onde há muitos anfitriões que precisam se conectar a um gateway comum **IPsec**. Uma central *certificate authority* (CA) é usada para assinar certificados RSA para hosts ou usuários. Esta central CA é responsável por transmitir confiança, incluindo as revogações de hosts ou usuários individuais.

- *NULL authentication* é usado para obter criptografia de malha sem autenticação. Ela protege contra ataques passivos, mas não protege contra ataques ativos. Entretanto, como **IKEv2** permite métodos de autenticação assimétricos, a autenticação NULL também pode ser usada para IPsec oportunista em escala de Internet, onde os clientes autenticam o servidor, mas os servidores não autenticam o cliente. Este modelo é similar aos sites seguros usando **TLS**.

Proteção contra computadores quânticos

Além destes métodos de autenticação, você pode usar o método *Postquantum Preshared Keys* (PPK) para se proteger contra possíveis ataques de computadores quânticos. Clientes individuais ou grupos de clientes podem usar seu próprio PPK especificando um (PPKID) que corresponde a uma chave pré-partilhada configurada fora da banda.

O uso do site **IKEv1** com as Chaves Pré-partilhadas forneceu proteção contra os atacantes quânticos. O redesenho de **IKEv2** não oferece esta proteção nativamente. **Libreswan** oferece o uso de *Postquantum Preshared Keys* (PPK) para proteger **IKEv2** conexões contra ataques quânticos.

Para ativar o suporte opcional PPK, adicione **ppk=yes** à definição da conexão. Para requerer o PPK, adicione **ppk=insist**. Então, cada cliente pode receber uma identificação PPK com um valor secreto que é comunicado fora da banda (e de preferência quantum safe). Os PPK's devem ser muito fortes em aleatoriedade e não devem ser baseados em palavras de dicionário. Os PPK ID e os próprios dados PPK são armazenados em **ipsec.secrets**, por exemplo:

```
@west @east : PPKS {\i1}"user1}" "thestringismeanttobearandomstr"
```

A opção **PPKS** refere-se aos PPKs estáticos. Uma função experimental utiliza PPKs dinâmicos baseados em um único painel. Em cada conexão, uma nova parte de um bloco é usada como PPK. Quando usado, essa parte do PPK dinâmico dentro do arquivo é sobrescrita com zeros para evitar a reutilização. Se não houver mais material de um único bloco, a conexão falha. Consulte a página de manual **ipsec.secrets(5)** para mais informações.

ATENÇÃO

A implementação de PPKs dinâmicos é fornecida como uma Pré-visualização Tecnológica, e esta funcionalidade deve ser usada com cautela.

3.8. IMPLEMENTAÇÃO DE UMA VPN IPSEC COMPATÍVEL COM FIPS

Use este procedimento para implantar uma solução VPN IPsec compatível com FIPS baseada em Libreswan. Os passos seguintes também permitem identificar quais algoritmos criptográficos estão disponíveis e quais estão desabilitados para Libreswan no modo FIPS.

Pré-requisitos

- O repositório **AppStream** está habilitado.

Procedimento

1. Instale os pacotes **libreswan**:

```
# yum install libreswan
```

2. Se você estiver reinstalando **Libreswan**, remova seu antigo banco de dados NSS:

```
# systemctl stop ipsec
# rm /etc/ipsec.d/*db
```

3. Iniciar o serviço **ipsec**, e permitir que o serviço seja iniciado automaticamente na inicialização:

```
# systemctl enable ipsec --now
```

4. Configure o firewall para permitir portas 500 e 4500/UDP para os protocolos IKE, ESP, e AH adicionando o serviço **ipsec**:

```
# firewall-cmd --add-service="ipsec"
# firewall-cmd --runtime-to-permanent
```

5. Mude o sistema para o modo FIPS no RHEL 8:

```
# fips-mode-setup --enable
```

6. Reinicie seu sistema para permitir que o kernel mude para o modo FIPS:

```
# reboot
```

Etapas de verificação

1. Para confirmar que Libreswan está funcionando no modo FIPS:

```
# ipsec whack --fipsstatus
000 FIPS mode enabled
```

2. Alternativamente, verifique as entradas para a unidade **ipsec** na revista **systemd**:

```
$ journalctl -u ipsec
...
Jan 22 11:26:50 localhost.localdomain pluto[3076]: FIPS Product: YES
```

```
Jan 22 11:26:50 localhost.localdomain pluto[3076]: FIPS Kernel: YES
Jan 22 11:26:50 localhost.localdomain pluto[3076]: FIPS Mode: YES
```

3. Para ver os algoritmos disponíveis no modo FIPS:

```
# ipsec pluto --selftest 2>&1 | head -11
FIPS Product: YES
FIPS Kernel: YES
FIPS Mode: YES
NSS DB directory: sql:/etc/ipsec.d
Initializing NSS
Opening NSS database "sql:/etc/ipsec.d" read-only
NSS initialized
NSS crypto library initialized
FIPS HMAC integrity support [enabled]
FIPS mode enabled for pluto daemon
NSS library is running in FIPS mode
FIPS HMAC integrity verification self-test passed
```

4. Para consultar algoritmos desabilitados no modo FIPS:

```
# ipsec pluto --selftest 2>&1 | grep disabled
Encryption algorithm CAMELLIA_CTR disabled; not FIPS compliant
Encryption algorithm CAMELLIA_CBC disabled; not FIPS compliant
Encryption algorithm SERPENT_CBC disabled; not FIPS compliant
Encryption algorithm TWOFISH_CBC disabled; not FIPS compliant
Encryption algorithm TWOFISH_SSH disabled; not FIPS compliant
Encryption algorithm NULL disabled; not FIPS compliant
Encryption algorithm CHACHA20_POLY1305 disabled; not FIPS compliant
Hash algorithm MD5 disabled; not FIPS compliant
PRF algorithm HMAC_MD5 disabled; not FIPS compliant
PRF algorithm AES_XCBC disabled; not FIPS compliant
Integrity algorithm HMAC_MD5_96 disabled; not FIPS compliant
Integrity algorithm HMAC_SHA2_256_TRUNCBUG disabled; not FIPS compliant
Integrity algorithm AES_XCBC_96 disabled; not FIPS compliant
DH algorithm MODP1024 disabled; not FIPS compliant
DH algorithm MODP1536 disabled; not FIPS compliant
DH algorithm DH31 disabled; not FIPS compliant
```

5. Para listar todos os algoritmos e cifras permitidas no modo FIPS:

```
# ipsec pluto --selftest 2>&1 | grep ESP | grep FIPS | sed "s/^.*FIPS//"
{256,192,*128}  aes_ccm, aes_ccm_c
{256,192,*128}  aes_ccm_b
{256,192,*128}  aes_ccm_a
[*192]  3des
{256,192,*128}  aes_gcm, aes_gcm_c
{256,192,*128}  aes_gcm_b
{256,192,*128}  aes_gcm_a
{256,192,*128}  aesctr
{256,192,*128}  aes
{256,192,*128}  aes_gmac
sha, sha1, sha1_96, hmac_sha1
sha512, sha2_512, sha2_512_256, hmac_sha2_512
sha384, sha2_384, sha2_384_192, hmac_sha2_384
```

```
sha2, sha256, sha2_256, sha2_256_128, hmac_sha2_256
aes_cmac
null
null, dh0
dh14
dh15
dh16
dh17
dh18
ecp_256, ecp256
ecp_384, ecp384
ecp_521, ecp521
```

Recursos adicionais

- Usando políticas criptográficas de todo o sistema

3.9. PROTEGENDO O BANCO DE DADOS IPSEC NSS POR UMA SENHA

Por padrão, o serviço IPsec cria seu banco de dados de Serviços de Segurança de Rede (NSS) com uma senha vazia durante a primeira inicialização. Adicione proteção por senha usando os seguintes passos.

NOTA

Nos lançamentos anteriores da RHEL até a versão 6.6, era necessário proteger o banco de dados IPsec NSS com uma senha para atender aos requisitos FIPS 140-2, pois as bibliotecas criptográficas NSS eram certificadas para o padrão FIPS 140-2 Nível 2. No RHEL 8, o NIST certificou o NSS para o Nível 1 desta norma, e este status não requer proteção por senha para o banco de dados.

Pré-requisito

- O diretório **/etc/ipsec.d** contém arquivos de banco de dados do NSS.

Procedimento

1. Ativar a proteção por senha para o banco de dados **NSS** para **Libreswan**:

```
# certutil -N -d sql:/etc/ipsec.d
Enter Password or Pin for "NSS Certificate DB":
Enter a password which will be used to encrypt your keys.
The password should be at least 8 characters long,
and should contain at least one non-alphabetic character.

Enter new password:
```

2. Crie o arquivo **/etc/ipsec.d/nsspassword** contendo a senha que você definiu na etapa anterior, por exemplo:

```
# cat /etc/ipsec.d/nsspassword
NSS Certificate DB:MyStrongPasswordHere
```

Observe que o arquivo **nsspassword** utiliza a seguinte sintaxe:

-

```
token_1_name:the_password
token_2_name:the_password
```

O token padrão do software NSS é **NSS Certificate DB**. Se seu sistema estiver rodando no modo FIPS, o nome do token é **NSS FIPS 140-2 Certificate DB**.

3. Dependendo de seu cenário, inicie ou reinicie o serviço **ipsec** depois de terminar o arquivo **nsspassword**:

```
# systemctl restart ipsec
```

Etapas de verificação

1. Verifique se o serviço **ipsec** está funcionando após você ter adicionado uma senha não vazia ao seu banco de dados NSS:

```
# systemctl status ipsec
● ipsec.service - Internet Key Exchange (IKE) Protocol Daemon for IPsec
  Loaded: loaded (/usr/lib/systemd/system/ipsec.service; enabled; vendor preset: disable>
  Active: active (running)...
```

2. Opcionalmente, verifique se o registro **Journal** contém entradas que confirmam uma inicialização bem sucedida:

```
# journalctl -u ipsec
...
pluto[23001]: NSS DB directory: sql:/etc/ipsec.d
pluto[23001]: Initializing NSS
pluto[23001]: Opening NSS database "sql:/etc/ipsec.d" read-only
pluto[23001]: NSS Password from file "/etc/ipsec.d/nsspassword" for token "NSS Certificate
DB" with length 20 passed to NSS
pluto[23001]: NSS crypto library initialized
...
```

Recursos adicionais

- A página do homem **certutil(1)**.
- Para mais informações sobre certificações relacionadas ao FIPS 140-2, consulte o artigo Base de Conhecimento de Normas Governamentais.

3.10. CONFIGURAÇÃO DE CONEXÕES IPSEC QUE OPTAM POR NÃO PARTICIPAR DAS POLÍTICAS DE CRIPTOGRAFIA DE TODO O SISTEMA

Políticas criptográficas para uma conexão em todo o sistema

As políticas criptográficas de todo o sistema RHEL criam uma conexão especial chamada **fault**. Esta conexão contém os valores padrão para as opções **ikev2**, **esp**, e **ike**. Entretanto, é possível anular os valores padrão especificando a opção mencionada no arquivo de configuração da conexão.

Por exemplo, a seguinte configuração permite conexões que usam IKEv1 com AES e SHA-1 ou SHA-2, e IPsec (ESP) com AES-GCM ou AES-CBC:

```
conn MyExample
  ...
  ikev2=never
  ike=aes-sha2,aes-sha1;modp2048
  esp=aes_gcm,aes-sha2,aes-sha1
  ...
```

Observe que o AES-GCM está disponível para IPsec (ESP) e para IKEv2, mas não para IKEv1.

Desabilitando políticas de criptografia de todo o sistema para todas as conexões

Para desativar políticas de criptografia em todo o sistema para todas as conexões IPsec, comente a seguinte linha no arquivo **/etc/ipsec.conf**:

```
incluir /etc/crypto-policies/back-ends/libreswan.config
```

Em seguida, adicione a opção **ikev2=never** ao seu arquivo de configuração de conexão.

Recursos adicionais

- Consulte Utilizando políticas criptográficas de todo o sistema para obter mais informações.

3.11. SOLUÇÃO DE PROBLEMAS EM CONFIGURAÇÕES DE VPN IPSEC

Os problemas relacionados às configurações de VPN IPsec ocorrem mais comumente devido a várias razões principais. Se você estiver encontrando tais problemas, você pode verificar se a causa do problema corresponde a algum dos seguintes cenários, e aplicar a solução correspondente.

Solução de problemas básicos de conexão

A maioria dos problemas com conexões VPN ocorre em novas implantações, onde os administradores configuraram pontos finais com opções de configuração não compatíveis. Além disso, uma configuração funcional pode de repente parar de funcionar, muitas vezes devido a valores incompatíveis recentemente introduzidos. Isto pode ser o resultado de um administrador mudar a configuração. Alternativamente, um administrador pode ter instalado uma atualização de firmware ou uma atualização de pacote com diferentes valores padrão para certas opções, tais como algoritmos de criptografia.

Para confirmar que uma conexão VPN IPsec é estabelecida:

```
# ipsec trafficstatus
006 #8: "vpn.example.com"[1] 192.0.2.1, type=ESP, add_time=1595296930, inBytes=5999,
outBytes=3231, id='@vpn.example.com', lease=100.64.13.5/32
```

Se a saída estiver vazia ou não mostrar uma entrada com o nome da conexão, o túnel está quebrado.

Para verificar se o problema está na conexão:

1. Recarregue a conexão *vpn.example.com*:

   ```
   # ipsec auto --add vpn.example.com
   002 added connection description "vpn.example.com"
   ```

2. A seguir, iniciar a conexão VPN:

   ```
   # ipsec auto --up vpn.example.com
   ```

Problemas relacionados a firewall-

O problema mais comum é que um firewall em um dos pontos terminais IPsec ou em um roteador entre os pontos terminais está soltando todos os pacotes do Internet Key Exchange (IKE).

- Para IKEv2, uma saída semelhante ao exemplo a seguir indica um problema com um firewall:

```
# ipsec auto --up vpn.example.com
181 "vpn.example.com"[1] 192.0.2.2 #15: initiating IKEv2 IKE SA
181 "vpn.example.com"[1] 192.0.2.2 #15: STATE_PARENT_I1: sent v2I1, expected v2R1
010 "vpn.example.com"[1] 192.0.2.2 #15: STATE_PARENT_I1: retransmission; will wait 0.5
seconds for response
010 "vpn.example.com"[1] 192.0.2.2 #15: STATE_PARENT_I1: retransmission; will wait 1
seconds for response
010 "vpn.example.com"[1] 192.0.2.2 #15: STATE_PARENT_I1: retransmission; will wait 2
seconds for
...
```

- Para o IKEv1, a saída do comando iniciador parece ser a mesma:

```
# ipsec auto --up vpn.example.com
002 "vpn.example.com" #9: initiating Main Mode
102 "vpn.example.com" #9: STATE_MAIN_I1: sent MI1, expecting MR1
010 "vpn.example.com" #9: STATE_MAIN_I1: retransmission; will wait 0.5 seconds for
response
010 "vpn.example.com" #9: STATE_MAIN_I1: retransmission; will wait 1 seconds for
response
010 "vpn.example.com" #9: STATE_MAIN_I1: retransmission; will wait 2 seconds for
response
...
```

Como o protocolo IKE, que é usado para configurar o IPsec, é criptografado, você pode solucionar apenas um subconjunto limitado de problemas usando a ferramenta **tcpdump**. Se um firewall estiver descartando pacotes IKE ou IPsec, você pode tentar encontrar a causa usando o utilitário **tcpdump**. Entretanto, **tcpdump** não pode diagnosticar outros problemas com conexões VPN IPsec.

- Para capturar a negociação da VPN e todos os dados criptografados na interface **eth0**:

```
# tcpdump -i eth0 -n -n esp or udp port 500 or udp port 4500 or tcp port 4500
```

Algoritmos, protocolos e políticas inadequados

As conexões VPN exigem que os pontos finais tenham algoritmos IKE, algoritmos IPsec e faixas de endereços IP correspondentes. Se ocorrer um descasamento, a conexão falha. Se você identificar um descasamento usando um dos seguintes métodos, conserte-o alinhando os algoritmos, protocolos ou políticas.

- Se o terminal remoto não estiver executando o IKE/IPsec, você pode ver um pacote ICMP indicando-o. Por exemplo, um pacote ICMP:

```
# ipsec auto --up vpn.example.com
...
000 "vpn.example.com"[1] 192.0.2.2 #16: ERROR: asynchronous network error report on
```

```
wlp2s0 (192.0.2.2:500), complainant 198.51.100.1: Connection refused [errno 111, origin
ICMP type 3 code 3 (not authenticated)]
...
```

- Exemplo de algoritmos IKE não compatíveis:

```
# ipsec auto --up vpn.example.com
...
003 "vpn.example.com"[1] 193.110.157.148 #3: dropping unexpected IKE_SA_INIT message
containing NO_PROPOSAL_CHOSEN notification; message payloads: N; missing payloads:
SA,KE,Ni
```

- Exemplo de algoritmos IPsec desajustados:

```
# ipsec auto --up vpn.example.com
...
182 "vpn.example.com"[1] 193.110.157.148 #5: STATE_PARENT_I2: sent v2I2, expected
v2R2 {auth=IKEv2 cipher=AES_GCM_16_256 integ=n/a prf=HMAC_SHA2_256
group=MODP2048}
002 "vpn.example.com"[1] 193.110.157.148 #6: IKE_AUTH response contained the error
notification NO_PROPOSAL_CHOSEN
```

  Uma versão não compatível do IKE também poderia resultar na queda do ponto final remoto sem resposta. Isto parece idêntico a um firewall que deixa cair todos os pacotes IKE.

- Exemplo de faixas de endereços IP inadequadas para IKEv2 (chamados Selecionadores de Tráfego - TS):

```
# ipsec auto --up vpn.example.com
...
1v2 "vpn.example.com" #1: STATE_PARENT_I2: sent v2I2, expected v2R2 {auth=IKEv2
cipher=AES_GCM_16_256 integ=n/a prf=HMAC_SHA2_512 group=MODP2048}
002 "vpn.example.com" #2: IKE_AUTH response contained the error notification
TS_UNACCEPTABLE
```

- Exemplo de faixas de endereços IP inadequadas para IKEv1:

```
# ipsec auto --up vpn.example.com
...
031 "vpn.example.com" #2: STATE_QUICK_I1: 60 second timeout exceeded after 0
retransmits.  No acceptable response to our first Quick Mode message: perhaps peer likes
no proposal
```

- Ao usar PreSharedKeys (PSK) no IKEv1, se ambos os lados não colocarem no mesmo PSK, toda a mensagem IKE se torna ilegível:

```
# ipsec auto --up vpn.example.com
...
003 "vpn.example.com" #1: received Hash Payload does not match computed value
223 "vpn.example.com" #1: sending notification INVALID_HASH_INFORMATION to
192.0.2.23:500
```

- No IKEv2, o erro de mismatched-PSK resulta em uma mensagem de AUTENTICATION_FAILED:

```
# ipsec auto --up vpn.example.com
...
002 "vpn.example.com" #1: IKE SA authentication request rejected by peer:
AUTHENTICATION_FAILED
```

Unidade máxima de transmissão

Além das firewalls bloqueando pacotes IKE ou IPsec, a causa mais comum de problemas de rede está relacionada ao aumento do tamanho dos pacotes criptografados. O hardware da rede fragmenta pacotes maiores que a unidade máxima de transmissão (MTU), por exemplo, 1500 bytes. Muitas vezes, os fragmentos são perdidos e os pacotes não conseguem se remontar. Isto leva a falhas intermitentes, quando um teste de ping, que usa pacotes de tamanho pequeno, funciona, mas o outro tráfego falha. Neste caso, você pode estabelecer uma sessão SSH, mas o terminal congela assim que é usado, por exemplo, inserindo o comando 'ls -al /usr' no host remoto.

Para contornar o problema, reduza o tamanho do MTU adicionando a opção **mtu=1400** ao arquivo de configuração do túnel.

Alternativamente, para conexões TCP, habilite uma regra iptables que altera o valor do MSS:

```
# iptables -I FORWARD -p tcp --tcp-flags SYN,RST SYN -j TCPMSS --clamp-mss-to-pmtu
```

Se o comando anterior não resolver o problema em seu cenário, especifique diretamente um tamanho menor no parâmetro **set-mss**:

```
# iptables -I FORWARD -p tcp --tcp-flags SYN,RST SYN -j TCPMSS --set-mss 1380
```

Tradução de endereços de rede (NAT)

Quando um host IPsec também serve como um roteador NAT, ele poderia acidentalmente refazer pacotes. O exemplo de configuração a seguir demonstra o problema:

```
conn myvpn
    left=172.16.0.1
    leftsubnet=10.0.2.0/24
    right=172.16.0.2
    rightsubnet=192.168.0.0/16
…
```

O sistema com o endereço 172.16.0.1 tem uma regra NAT:

```
iptables -t nat -I POSTROUTING -o eth0 -j MASQUERADE
```

Se o sistema no endereço 10.0.2.33 envia um pacote para 192.168.0.1, então o roteador traduz a fonte 10.0.2.33 para 172.16.0.1 antes de aplicar a criptografia IPsec.

Então, o pacote com o endereço de origem 10.0.2.33 não corresponde mais à configuração **conn myvpn**, e o IPsec não encripta este pacote.

Para resolver este problema, insira neste exemplo regras que excluam NAT para faixas de sub-rede IPsec de destino no roteador:

```
iptables -t nat -I POSTROUTING -s 10.0.2.0/24 -d 192.168.0.0/16 -j RETORNO
```

Bugs do subsistema IPsec do Kernel

O subsistema IPsec do kernel pode falhar, por exemplo, quando um bug causa uma dessincronização do espaço do usuário IKE e do kernel IPsec. Para verificar a existência de tais problemas:

```
$ cat /proc/net/xfrm_stat
XfrmInError               0
XfrmInBufferError         0
...
```

Qualquer valor não nulo na saída do comando anterior indica um problema. Se você encontrar este problema, abra um novo caso de suporte e anexe a saída do comando anterior junto com os logs correspondentes do IKE.

Toras de Libreswan

Libreswan registra usando o protocolo **syslog** por padrão. Você pode usar o comando **journalctl** para encontrar entradas de log relacionadas ao IPsec. Como as entradas correspondentes ao log são enviadas pelo daemon **pluto** IKE, procure a palavra-chave "pluto", por exemplo:

```
$ journalctl -b | grep pluto
```

Para mostrar um registro ao vivo para o serviço **ipsec**:

```
$ journalctl -f -u ipsec
```

Se o nível padrão de registro não revelar seu problema de configuração, habilite os registros de depuração adicionando a opção **plutodebug=all** à seção **config setup** no arquivo **/etc/ipsec.conf**.

Observe que o registro de depuração produz muitas entradas, e é possível que a taxa de serviço **journald** ou **syslogd** limite as mensagens **syslog**. Para garantir que você tenha registros completos, redirecione o registro para um arquivo. Edite o **/etc/ipsec.conf**, e adicione o **logfile=/var/log/pluto.log** na seção **config setup**.

Recursos adicionais

- Solução de problemas usando arquivos de log
- Usando e configurando o firewalld
- **tcpdump(8)** e **ipsec.conf(5)** páginas man

3.12. INFORMAÇÕES RELACIONADAS

Os seguintes recursos fornecem informações adicionais sobre **Libreswan** e o daemon **ipsec**.

Documentação instalada

- **ipsec(8)** página man - Descreve opções de comando para **ipsec**.
- **ipsec.conf(5)** página man - Contém informações sobre a configuração **ipsec**.
- **ipsec.secrets(5)** man page - Descreve o formato do arquivo **ipsec.secrets**.

- **ipsec_auto(8)** man page - Descreve o uso do cliente de linha de comando **auto** para manipular as conexões IPsec de Libreswan estabelecidas através de trocas automáticas de chaves.
- **ipsec_rsasigkey(8)** man page - Descreve a ferramenta utilizada para gerar chaves de assinatura RSA.
- **/usr/share/doc/libreswan-*version*/**

Documentação on-line

https://libreswan.org

O site do projeto upstream.

https://libreswan.org/wiki

O Wiki do Projeto Libreswan.

https://libreswan.org/man/

Todas as páginas de Libreswan man.

Publicação Especial NIST 800-77: Guia para VPNs IPsec

Orientação prática para organizações na implementação de serviços de segurança baseados em IPsec.

CAPÍTULO 4. CONFIGURAÇÃO DE MACSEC

A seção a seguir fornece informações sobre como configurar **Media Control Access Security** (**MACsec**), que é uma tecnologia de segurança padrão 802.1AE IEEE para comunicação segura em todo o tráfego em links Ethernet.

4.1. INTRODUÇÃO AO MACSEC

Media Access Control Security (**MACsec**, IEEE 802.1AE) codifica e autentica todo o tráfego em LANs com o algoritmo GCM-AES-128. **MACsec** pode proteger não apenas **IP** mas também o Protocolo de Resolução de Endereços (ARP), Neighbor Discovery (ND), ou **DHCP**. Enquanto **IPsec** opera na camada de rede (camada 3) e **SSL** ou **TLS** na camada de aplicação (camada 7), **MACsec** opera na camada de link de dados (camada 2). Combine **MACsec** com protocolos de segurança para outras camadas de rede para tirar proveito das diferentes características de segurança que estes padrões oferecem.

4.2. USANDO MACSEC COM A FERRAMENTA NMCLI

Este procedimento mostra como configurar **MACsec** com a ferramenta **nmcli**.

Pré-requisitos

- O **NetworkManager** deve estar funcionando.
- Você já tem um CAK hexadecimal de 16 bytes (**$MKA_CAK**) e um CKN hexadecimal de 32 bytes (**$MKA_CKN**).

Procedimento

1. Para adicionar uma nova conexão usando **nmcli**, entre:

```
~]# nmcli connection add type macsec \
 con-name test-macsec+ ifname macsec0 \
 connection.autoconnect no \
 macsec.parent enp1s0 macsec.mode psk \
 macsec.mka-cak $MKA_CAK \
 macsec.mka-ckn $MKA_CKN
```

   Substitua *macsec0* pelo nome do dispositivo que você deseja configurar.

2. Para ativar a conexão, entre:

```
~]# nmcli conexão up test-macsec
```

Após esta etapa, o dispositivo *macsec0* é configurado e pode ser utilizado para a criação de redes.

4.3. USANDO MACSEC COM WPA_SUPPLICANT

Este procedimento mostra como habilitar **MACsec** com um switch que realiza a autenticação usando um par pré-partilhado de Chave de Conectividade de Associação/CAK Name (CAK/CKN).

Procedimento

1. Criar um par CAK/CKN. Por exemplo, o seguinte comando gera uma chave de 16 bytes em notação hexadecimal:

```
~]$ dd if=/dev/urandom count=16 bs=1 2> /dev/null | hexdump -e '1/2 "%02x"'
```

2. Crie o arquivo de configuração **wpa_supplicant.conf** e acrescente as seguintes linhas a ele:

```
ctrl_interface=/var/run/wpa_supplicant
eapol_version=3
ap_scan=0
fast_reauth=1

network={
    key_mgmt=NONE
    eapol_flags=0
    macsec_policy=1

    mka_cak=0011... # 16 bytes hexadecimal
    mka_ckn=2233... # 32 bytes hexadecimal
}
```

   Use os valores da etapa anterior para completar as linhas **mka_cak** e **mka_ckn** no arquivo de configuração **wpa_supplicant.conf**.

   Para mais informações, consulte a página de manual **wpa_supplicant.conf(5)**.

3. Assumindo que você está usando *wlp61s0* para se conectar à sua rede, comece **wpa_supplicant** usando o seguinte comando:

```
~]# wpa_supplicant -i wlp61s0 -Dmacsec_linux -c wpa_supplicant.conf
```

4.4. INFORMAÇÕES RELACIONADAS

Para mais detalhes, veja o artigo O que há de novo em MACsec: criação de MACsec usando wpa_supplicant e (opcionalmente) NetworkManager. Além disso, veja o artigo MACsec: uma solução diferente para criptografar o tráfego de rede para mais informações sobre a arquitetura de uma rede **MACsec**, cenários de caso de uso e exemplos de configuração.

CAPÍTULO 5. USANDO E CONFIGURANDO O FIREWALLD

A *firewall* é uma forma de proteger as máquinas de qualquer tráfego indesejado do exterior. Ele permite aos usuários controlar o tráfego de entrada da rede nas máquinas host, definindo um conjunto de *firewall rules*. Estas regras são usadas para ordenar o tráfego de entrada e ou bloqueá-lo ou permitir a passagem.

Note que **firewalld** com **nftables** backend não suporta a passagem das regras personalizadas **nftables** para **firewalld**, usando a opção **--direct**.

5.1. QUANDO USAR FIREWALLD, NFTABLES, OU IPTABLES

A seguir, uma breve visão geral em que cenário você deve utilizar uma das seguintes utilidades:

- **firewalld**: Use o utilitário **firewalld** para casos simples de uso de firewall. O utilitário é fácil de usar e cobre os casos de uso típico para estes cenários.
- **nftables**: Use o utilitário **nftables** para criar firewalls complexos e de desempenho crítico, como para toda uma rede.
- **iptables**: O utilitário **iptables** no Red Hat Enterprise Linux 8 usa a API do kernel **nf_tables** ao invés do back end **legacy**. A API **nf_tables** fornece compatibilidade retroativa para que scripts que usam os comandos **iptables** ainda funcionem no Red Hat Enterprise Linux 8. Para novos scripts de firewall, a Red Hat recomenda usar **nftables**.

IMPORTANTE

Para evitar que os diferentes serviços de firewall influenciem uns aos outros, execute apenas um deles em um host RHEL, e desabilite os outros serviços.

5.2. COMEÇANDO COM FIREWALLD

5.2.1. firewalld

firewalld é um daemon de serviço de firewall que fornece um firewall dinâmico personalizável baseado em host com uma interface **D-Bus**. Sendo dinâmico, ele permite criar, alterar e apagar as regras sem a necessidade de reiniciar o daemon de firewall cada vez que as regras são alteradas.

firewalld utiliza os conceitos de *zones* e *services*, que simplificam a gestão do tráfego. As zonas são conjuntos de regras pré-definidas. As interfaces e fontes de rede podem ser atribuídas a uma zona. O tráfego permitido depende da rede à qual seu computador está conectado e do nível de segurança que esta rede é atribuída. Os serviços de firewall são regras predefinidas que cobrem todas as configurações necessárias para permitir o tráfego de entrada para um serviço específico e se aplicam dentro de uma zona.

Os serviços utilizam um ou mais *ports* ou *addresses* para comunicação em rede. Os firewalls filtram a comunicação com base em portas. Para permitir o tráfego de rede para um serviço, suas portas devem ser *open*. **firewalld** bloqueia todo o tráfego nas portas que não estão explicitamente definidas como abertas. Algumas zonas, tais como *trusted*, permitem todo o tráfego por padrão.

Recursos adicionais

- **firewalld(1)** página do homem

5.2.2. Zonas

firewalld pode ser usado para separar as redes em diferentes zonas de acordo com o nível de confiança que o usuário decidiu colocar nas interfaces e no tráfego dentro daquela rede. Uma conexão só pode ser parte de uma zona, mas uma zona pode ser usada para muitas conexões de rede.

NetworkManager notifica **firewalld** sobre a zona de uma interface. Você pode atribuir zonas para interfaces com:

- **NetworkManager**
- ferramenta**firewall-config**
- **firewall-cmd** ferramenta de linha de comando
- O console web RHEL

Os três últimos só podem editar os arquivos de configuração **NetworkManager** apropriados. Se você mudar a zona da interface usando o console web, **firewall-cmd** ou **firewall-config**, o pedido é encaminhado para **NetworkManager** e não é tratado por **firewalld**.

As zonas pré-definidas são armazenadas no diretório **/usr/lib/firewalld/zones/** e podem ser aplicadas instantaneamente a qualquer interface de rede disponível. Estes arquivos são copiados para o diretório **/etc/firewalld/zones/** somente após serem modificados. As configurações padrão das zonas pré-definidas são as seguintes:

block

Qualquer conexão de rede que chegue é rejeitada com uma mensagem proibida para **IPv4** e para **IPv6**. Somente conexões de rede iniciadas de dentro do sistema são possíveis.

dmz

Para computadores em sua zona desmilitarizada que são de acesso público com acesso limitado à sua rede interna. Somente conexões de entrada selecionadas são aceitas.

drop

Qualquer pacote de rede recebido é descartado sem nenhuma notificação. Somente as conexões de rede de saída são possíveis.

external

Para uso em redes externas com mascaramento habilitado, especialmente para roteadores. Você não confia nos outros computadores da rede para não danificar seu computador. Somente conexões de entrada selecionadas são aceitas.

home

Para uso em casa quando você confia principalmente nos outros computadores da rede. Somente as conexões de entrada selecionadas são aceitas.

internal

Para uso em redes internas quando você confia principalmente nos outros computadores da rede. Somente as conexões de entrada selecionadas são aceitas.

public

Para uso em áreas públicas onde você não confia em outros computadores na rede. Somente conexões de entrada selecionadas são aceitas.

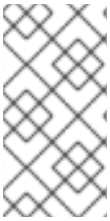
trusted

Todas as conexões de rede são aceitas.

work

Para uso no trabalho, onde você confia principalmente nos outros computadores da rede. Somente as conexões de entrada selecionadas são aceitas.

Uma dessas zonas está definida como a zona *default*. Quando as conexões de interface são adicionadas a **NetworkManager**, elas são atribuídas à zona padrão. Na instalação, a zona padrão em **firewalld** é definida como a zona **public**. A zona padrão pode ser alterada.



NOTA

Os nomes das zonas de rede devem ser auto-explicativos e permitir que os usuários tomem rapidamente uma decisão razoável. Para evitar quaisquer problemas de segurança, revisar a configuração padrão da zona e desativar quaisquer serviços desnecessários de acordo com suas necessidades e avaliações de risco.

Recursos adicionais

- **firewalld.zone(5)** página do homem

5.2.3. Serviços pré-definidos

Um serviço pode ser uma lista de portas locais, protocolos, portas de origem e destinos, bem como uma lista de módulos de ajuda de firewall carregados automaticamente se um serviço for ativado. O uso de serviços economiza tempo dos usuários porque eles podem realizar várias tarefas, tais como abrir portas, definir protocolos, permitir o envio de pacotes e mais, em uma única etapa, em vez de configurar tudo, um após o outro.

As opções de configuração de serviço e informações genéricas do arquivo estão descritas na página de manual **firewalld.service(5)**. Os serviços são especificados por meio de arquivos de configuração XML individuais, que são nomeados no formato a seguir ***service-name*.xml**. Os nomes dos protocolos são preferidos aos nomes dos serviços ou aplicativos em **firewalld**.

Os serviços podem ser adicionados e removidos usando a ferramenta gráfica **firewall-config**, **firewall-cmd**, e **firewall-offline-cmd**.

Alternativamente, você pode editar os arquivos XML no diretório **/etc/firewalld/services/**. Se um serviço não for adicionado ou alterado pelo usuário, então nenhum arquivo XML correspondente é encontrado em **/etc/firewalld/services/**. Os arquivos no diretório **/usr/lib/firewalld/services/** podem ser usados como modelos se você quiser adicionar ou alterar um serviço.

Recursos adicionais

- **firewalld.service(5)** página do homem

5.3. INSTALANDO A FERRAMENTA DE CONFIGURAÇÃO FIREWALL-CONFIG GUI

Para usar a ferramenta de configuração **firewall-config** GUI, instale o pacote **firewall-config**.

Procedimento

1. Digite o seguinte comando como **root**:

```
# yum instalar firewall-configurar
```

Alternativamente, em **GNOME, use the Super key and type `Software** para lançar o aplicativo **Software Sources**. Digite **firewall** na caixa de busca, que aparece após selecionar o botão de busca no canto superior direito. Selecione o item **Firewall** nos resultados da busca e clique no botão **Instalar**.

2. Para executar **firewall-config**, use o comando **firewall-config** ou pressione a tecla **Super** para entrar no **Activities Overview**, digite **firewall**, e pressione **Enter**.

5.4. VISUALIZANDO O STATUS ATUAL E AS CONFIGURAÇÕES DE FIREWALLD

5.4.1. Visualizando o status atual de firewalld

O serviço de firewall, **firewalld**, é instalado no sistema por padrão. Use a interface **firewalld** CLI para verificar se o serviço está sendo executado.

Procedimento

1. Para ver o status do serviço:

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

2. Para mais informações sobre o status do serviço, use o sub-comando **systemctl status**:

```
# 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
```

Recursos adicionais

É importante saber como **firewalld** está configurado e quais regras estão em vigor antes de tentar editar as configurações. Para exibir as configurações do firewall, veja Seção 5.4.2, “Visualizando os ajustes firewalld atuais”

5.4.2. Visualizando os ajustes firewalld atuais

5.4.2.1. Visualização de serviços permitidos usando GUI

Para visualizar a lista de serviços utilizando o gráfico **firewall-config** pressione a tecla **Super** para entrar na Visão Geral das Atividades, digite **firewall**, e pressione **Enter**. O **firewall-config** aparece a ferramenta. Agora você pode visualizar a lista de serviços na guia **Services**.

Alternativamente, para iniciar a ferramenta gráfica de configuração de firewall usando a linha de comando, digite o seguinte comando:

```
$ firewall-config
```

A janela **Firewall Configuration** se abre. Note que este comando pode ser executado como um usuário normal, mas ocasionalmente você é solicitado a obter uma senha de administrador.

5.4.2.2. Visualizando as configurações firewalld usando CLI

Com o cliente CLI, é possível obter diferentes visões das configurações atuais do firewall. A opção **--list-all** mostra uma visão completa das configurações do **firewalld**.

firewalld utiliza zonas para gerenciar o tráfego. Se uma zona não for especificada pela opção **--zone**, o comando é efetivo na zona padrão atribuída à interface de rede ativa e à conexão.

Para listar todas as informações relevantes para a zona padrão:

```
# 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:
```

Para especificar a zona para a qual devem ser exibidas as configurações, acrescente o **--zone=*zone-name*** argumento para o comando **firewall-cmd --list-all**, por exemplo:

```
# 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]
```

Para ver as configurações para determinadas informações, tais como serviços ou portos, use uma opção específica. Veja as páginas do manual **firewalld** ou obtenha uma lista das opções usando a ajuda do comando:

```
# firewall-cmd --help

Usage: firewall-cmd [OPTIONS...]

General Options
  -h, --help          Prints a short help text and exists
  -V, --version       Print the version string of firewalld
  -q, --quiet         Do not print status messages

Status Options
```

```
  --state             Return and print firewalld state
  --reload            Reload firewall and keep state information
... [trimmed for clarity]
```

Por exemplo, para ver quais serviços são permitidos na zona atual:

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

NOTA

Listar as configurações para uma determinada subparte usando a ferramenta CLI pode, às vezes, ser difícil de interpretar. Por exemplo, você permite o serviço **SSH** e **firewalld** abre a porta necessária (22) para o serviço. Mais tarde, se você listar os serviços permitidos, a lista mostra o serviço **SSH**, mas se você listar as portas abertas, ela não mostra nenhuma. Portanto, recomenda-se usar a opção **--list-all** para garantir que você receba uma informação completa.

5.5. INICIANDO O FIREWALLD

Procedimento

1. Para iniciar **firewalld**, digite o seguinte comando como **root**:

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

2. Para garantir que **firewalld** comece automaticamente no início do sistema, digite o seguinte comando como **root**:

```
# systemctl habilita firewalld
```

5.6. PARANDO A FIREWALLD

Procedimento

1. Para parar **firewalld**, digite o seguinte comando como **root**:

```
# systemctl stop firewalld
```

2. Para evitar que o **firewalld** comece automaticamente no início do sistema:

```
# systemctl desativar firewalld
```

3. Para garantir que o firewalld não seja iniciado, acesse a interface **firewalld D-Bus** e também se outros serviços exigirem **firewalld**:

```
# Systemctl máscara firewalld
```

5.7. TEMPO DE EXECUÇÃO E AJUSTES PERMANENTES

Quaisquer mudanças comprometidas no modo *runtime* só se aplicam enquanto **firewalld** estiver em funcionamento. Quando **firewalld** é reiniciado, as configurações revertem para seus valores *permanent*.

Para tornar as mudanças persistentes através de reinicializações, aplicá-las novamente usando a opção **--permanent**. Alternativamente, para fazer alterações persistentes enquanto **firewalld** estiver em execução, use a opção **--runtime-to-permanent firewall-cmd** .

Se você definir as regras enquanto **firewalld** estiver funcionando usando apenas a opção **--permanent**, elas não se tornam efetivas antes de **firewalld** ser reiniciado. Entretanto, reiniciar **firewalld** fecha todas as portas abertas e pára o tráfego da rede.

Modificando configurações em tempo de execução e configuração permanente usando CLI

Usando o CLI, você não modifica as configurações do firewall em ambos os modos ao mesmo tempo. Você modifica apenas o tempo de execução ou o modo permanente. Para modificar as configurações do firewall no modo permanente, use a opção **--permanent** com o comando **firewall-cmd**.

```
# firewall-cmd --permanente <outras opções>
```

Sem esta opção, o comando modifica o modo de tempo de execução.

Para alterar as configurações em ambos os modos, você pode usar dois métodos:

1. Alterar as configurações de tempo de execução e depois torná-las permanentes como a seguir:

```
# firewall-cmd <other options>
# firewall-cmd --runtime-to-permanent
```

2. Definir configurações permanentes e recarregar as configurações no modo tempo de execução:

```
# firewall-cmd --permanent <other options>
# firewall-cmd --reload
```

O primeiro método permite testar as configurações antes de aplicá-las no modo permanente.

NOTA

É possível, especialmente em sistemas remotos, que uma configuração incorreta resulte em um bloqueio do usuário fora de uma máquina. Para evitar tais situações, use a opção **--timeout**. Após um determinado período de tempo, qualquer mudança reverte para seu estado anterior. O uso desta opção exclui a opção **--permanent**.

Por exemplo, para adicionar o serviço **SSH** por 15 minutos:

```
# firewall-cmd --add-service=ssh --timeout 15m
```

5.8. VERIFICAÇÃO DA CONFIGURAÇÃO FIREWALLD PERMANENTE

Em certas situações, por exemplo, após editar manualmente os arquivos de configuração **firewalld**, os administradores querem verificar se as mudanças estão corretas. Esta seção descreve como verificar a configuração permanente do serviço **firewalld**.

Pré-requisitos

- O serviço **firewalld** está funcionando.

Procedimento

1. Verificar a configuração permanente do serviço **firewalld**:

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

Se a configuração permanente for válida, o comando retorna **success**. Em outros casos, o comando retorna um erro com mais detalhes, tais como os seguintes:

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

5.9. CONTROLE DO TRÁFEGO DA REDE USANDO FIREWALLD

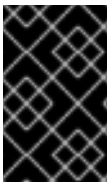
5.9.1. Desabilitação de todo o tráfego em caso de emergência usando CLI

Em uma situação de emergência, como um ataque ao sistema, é possível desativar todo o tráfego da rede e cortar o atacante.

Procedimento

1. Para desativar imediatamente o tráfego em rede, ligue o modo de pânico:

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



IMPORTANTE

A ativação do modo de pânico interrompe todo o tráfego em rede. Por este motivo, ele deve ser usado somente quando você tiver acesso físico à máquina ou se estiver logado usando um console serial.

A desativação do modo de pânico reverte o firewall para suas configurações permanentes. Para desativar o modo de pânico:

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

Para ver se o modo de pânico está ligado ou desligado, use:

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

5.9.2. Controle de tráfego com serviços pré-definidos usando CLI

O método mais simples para controlar o tráfego é adicionar um serviço pré-definido a **firewalld**. Isto abre todas as portas necessárias e modifica outras configurações de acordo com o *service definition file* .

Procedimento

1. Verifique se o serviço já não é permitido:

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

2. Liste todos os serviços pré-definidos:

```
# 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. Acrescente o serviço aos serviços permitidos:

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

4. Faça com que as novas configurações sejam persistentes:

```
# Firewall-cmd - tempo de execução a permanente
```

5.9.3. Controle de tráfego com serviços pré-definidos usando GUI

Para ativar ou desativar um serviço pré-definido ou personalizado:

1. Comece o **firewall-config** e selecionar a zona de rede cujos serviços devem ser configurados.
2. Selecione a guia **Services**.
3. Selecione a caixa de seleção para cada tipo de serviço em que você deseja confiar ou desmarque a caixa de seleção para bloquear um serviço.

Para editar um serviço:

1. Comece o **firewall-config** ferramenta.
2. Selecione **Permanent** a partir do menu etiquetado **Configuration**. Ícones adicionais e botões de menu aparecem na parte inferior da janela **Serviços**.
3. Selecione o serviço que você deseja configurar.

As guias **Ports**, **Protocols**, e **Source Port** permitem adicionar, alterar e remover portas, protocolos e porta de origem para o serviço selecionado. A aba módulos é para configurar **Netfilter** módulos auxiliares. A aba **Destination** permite limitar o tráfego a um endereço de destino específico e ao Protocolo Internet (**IPv4** ou **IPv6**).

NOTA

Não é possível alterar as configurações de serviço no modo **Runtime**.

5.9.4. Adicionando novos serviços

Os serviços podem ser adicionados e removidos usando o gráfico **firewall-config** **firewall-cmd** , e **firewall-offline-cmd**. Alternativamente, você pode editar os arquivos XML em **/etc/firewalld/services/**. Se um serviço não for adicionado ou alterado pelo usuário, então nenhum arquivo XML correspondente

é encontrado em **/etc/firewalld/services/**. Os arquivos **/usr/lib/firewalld/services/** podem ser usados como modelos se você quiser adicionar ou alterar um serviço.

NOTA

Os nomes dos serviços devem ser alfanuméricos e podem, adicionalmente, incluir apenas os caracteres _ (sublinhado) e **-** (traço).

Procedimento

Para adicionar um novo serviço em um terminal, use **firewall-cmd**, ou **firewall-offline-cmd** no caso de não estar ativo **firewalld**.

1. Digite o seguinte comando para adicionar um serviço novo e vazio:

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

2. Para adicionar um novo serviço usando um arquivo local, use o seguinte comando:

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

   Você pode mudar o nome do serviço com o **--name=*service-name*** opção.

3. Assim que as configurações do serviço são alteradas, uma cópia atualizada do serviço é colocada em **/etc/firewalld/services/**.
   Como **root**, você pode digitar o seguinte comando para copiar um serviço manualmente:

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

firewalld carrega arquivos de **/usr/lib/firewalld/services** em primeiro lugar. Se os arquivos forem colocados em **/etc/firewalld/services** e forem válidos, então estes substituirão os arquivos correspondentes de **/usr/lib/firewalld/services**. Os arquivos anulados em **/usr/lib/firewalld/services** são usados assim que os arquivos correspondentes em **/etc/firewalld/services** forem removidos ou se **firewalld** tiver sido solicitado a carregar os padrões dos serviços. Isto se aplica somente ao ambiente permanente. Uma recarga é necessária para obter estas falhas também no ambiente de tempo de execução.

5.9.5. Controle de portos usando CLI

Os portos são dispositivos lógicos que permitem a um sistema operacional receber e distinguir o tráfego da rede e encaminhá-lo de acordo com os serviços do sistema. Estes são normalmente representados por um daemon que escuta no porto, ou seja, espera por qualquer tráfego que chegue a este porto.

Normalmente, os serviços de sistema escutam nos portos padrão que lhes são reservados. O daemon **httpd**, por exemplo, ouve no porto 80. Entretanto, os administradores de sistema, por padrão, configuram daemons para ouvir em diferentes portas para aumentar a segurança ou por outras razões.

5.9.5.1. Abertura de um porto

Através de portas abertas, o sistema é acessível do exterior, o que representa um risco de segurança. Geralmente, mantenha as portas fechadas e só as abra se elas forem necessárias para determinados serviços.

Procedimento

Para obter uma lista de portos abertos na zona atual:

1. Liste todos os portos permitidos:

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

2. Adicione uma porta aos portos permitidos para abri-la para o tráfego de entrada:

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

3. Faça com que as novas configurações sejam persistentes:

```
# Firewall-cmd - tempo de execução a permanente
```

Os tipos de portos são **tcp**, **udp**, **sctp**, ou **dccp**. O tipo deve corresponder ao tipo de comunicação em rede.

5.9.5.2. Fechamento de um porto

Quando uma porta aberta não for mais necessária, feche essa porta em **firewalld**. É altamente recomendável fechar todas as portas desnecessárias assim que elas não forem utilizadas, pois deixar uma porta aberta representa um risco à segurança.

Procedimento

Para fechar um porto, removê-lo da lista de portos permitidos:

1. Liste todos os portos permitidos:

```
# firewall-cmd --list-ports
[WARNING]
====
This command will only give you a list of ports that have been opened as ports. You will not be able to see any open ports that have been opened as a service. Therefore, you should consider using the --list-all option instead of --list-ports.
====
```

2. Retirar o porto dos portos permitidos para fechá-lo para o tráfego de entrada:

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

3. Faça com que as novas configurações sejam persistentes:

```
# Firewall-cmd - tempo de execução a permanente
```

5.9.6. Abertura de portos usando GUI

Para permitir o tráfego através do firewall até uma determinada porta:

1. Comece o **firewall-config** e selecione a zona de rede cujas configurações você deseja alterar.

2. Selecione a aba **Ports** e clique no botão **Adicionar**, no lado direito. A janela **Port and Protocol** se abre.

3. Digite o número da porta ou intervalo de portas a permitir.

4. Selecione **tcp** ou **udp** a partir da lista.

5.9.7. Controle de tráfego com protocolos usando GUI

Para permitir o tráfego através do firewall usando um determinado protocolo:

1. Comece o **firewall-config** e selecione a zona de rede cujas configurações você deseja alterar.

2. Selecione a aba **Protocols** e clique no botão **Add** no lado direito. A janela **Protocol** se abre.

3. Selecione um protocolo da lista ou selecione a caixa de seleção **Other Protocol** e digite o protocolo no campo.

5.9.8. Abertura de portas de origem usando GUI

Permitir o tráfego através do firewall a partir de uma determinada porta:

1. Inicie a ferramenta de configuração de firewall e selecione a zona de rede cujas configurações você deseja alterar.

2. Selecione a aba **Source Port** e clique no botão **Add** no lado direito. A janela **Source Port** se abre.

3. Digite o número da porta ou intervalo de portas a permitir. Selecione **tcp** ou **udp** da lista.

5.10. TRABALHANDO COM ZONAS FIREWALLD

As zonas representam um conceito para gerenciar o tráfego de entrada de forma mais transparente. As zonas são conectadas a interfaces de rede ou designadas a uma gama de endereços de origem. As regras de firewall são gerenciadas independentemente para cada zona, o que permite definir configurações complexas de firewall e aplicá-las ao tráfego.

5.10.1. Listagem de zonas

Procedimento

1. Para ver quais zonas estão disponíveis em seu sistema:

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

   O comando **firewall-cmd --get-zones** exibe todas as zonas que estão disponíveis no sistema, mas não mostra nenhum detalhe para zonas específicas.

2. Para ver informações detalhadas para todas as zonas:

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

3. Para ver informações detalhadas para uma zona específica:

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

5.10.2. Modificação de configurações firewalld para uma determinada zona

Os sites Seção 5.9.2, “Controle de tráfego com serviços pré-definidos usando CLI” e Seção 5.9.5, “Controle de portos usando CLI” explicam como adicionar serviços ou modificar portos no escopo da zona de trabalho atual. S vezes, é necessário estabelecer regras em uma zona diferente.

Procedimento

1. Para trabalhar em uma zona diferente, use o **--zone=*zone-name*** opção. Por exemplo, para permitir o serviço **SSH** na zona *public*:

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

5.10.3. Mudando a zona padrão

Os administradores de sistema atribuem uma zona a uma interface de rede em seus arquivos de configuração. Se uma interface não for atribuída a uma zona específica, ela será atribuída à zona padrão. Após cada reinício do serviço **firewalld**, **firewalld** carrega as configurações para a zona padrão e a torna ativa.

Procedimento

Para configurar a zona padrão:

1. Exibir a zona padrão atual:

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

2. Defina a nova zona padrão:

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

NOTA

Seguindo este procedimento, a configuração é permanente, mesmo sem a opção **--permanent**.

5.10.4. Atribuição de uma interface de rede a uma zona

É possível definir diferentes conjuntos de regras para diferentes zonas e, em seguida, alterar as configurações rapidamente alterando a zona da interface que está sendo utilizada. Com várias interfaces, uma zona específica pode ser definida para cada uma delas para distinguir o tráfego que está passando por elas.

Procedimento

Para atribuir a zona a uma interface específica:

1. Relacione as zonas ativas e as interfaces atribuídas a elas:

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

2. Atribuir a interface a uma zona diferente:

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

5.10.5. Atribuição de uma zona a uma conexão usando nmcli

Este procedimento descreve como adicionar uma zona firewalld a uma conexão NetworkManager usando o utilitário **nmcli**.

Procedimento

1. Atribuir a zona ao perfil de conexão do NetworkManager:

```
# nmcli conexão modificar profile connection.zone zone_name
```

2. Recarregue a conexão:

```
# nmcli conexão acima profile
```

5.10.6. Atribuição manual de uma zona a uma conexão de rede em um arquivo ifcfg

Quando a conexão é gerenciada por **NetworkManager**deve estar ciente de uma zona que utiliza. Para cada conexão de rede, uma zona pode ser especificada, o que proporciona a flexibilidade de várias configurações de firewall de acordo com a localização do computador com dispositivos portáteis. Assim, as zonas e configurações podem ser especificadas para diferentes locais, como empresa ou residência.

Procedimento

1. Para definir uma zona para uma conexão, edite o **/etc/sysconfig/network-scripts/ifcfg-*connection_name*** e acrescentar uma linha que atribua uma zona a esta conexão:

```
ZONA=zone_name
```

5.10.7. Criando uma nova zona

Para usar zonas personalizadas, criar uma nova zona e usá-la como uma zona pré-definida. Novas zonas requerem a opção **--permanent**, caso contrário o comando não funciona.

Procedimento

Para criar uma nova zona:

1. Criar uma nova zona:

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

2. Verifique se a nova zona é adicionada a seus ajustes permanentes:

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

3. Faça com que as novas configurações sejam persistentes:

```
# Firewall-cmd - tempo de execução a permanente
```

5.10.8. Arquivos de configuração de zona

Zonas também podem ser criadas usando um *zone configuration file*. Esta abordagem pode ser útil quando você precisa criar uma nova zona, mas quer reutilizar as configurações de uma zona diferente e apenas alterá-las um pouco.

Um arquivo de configuração de zona **firewalld** contém as informações para uma zona. Estas são a descrição da zona, serviços, portos, protocolos, icmp-blocks, mascarada, forward-ports e regras de linguagem rica em um formato de arquivo XML. O nome do arquivo tem que ser ***zone-name*.xml** onde o comprimento de *zone-name* é atualmente limitado a 17 caracteres. Os arquivos de configuração da zona estão localizados nos diretórios **/usr/lib/firewalld/zones/** e **/etc/firewalld/zones/**.

O exemplo a seguir mostra uma configuração que permite um serviço (**SSH**) e uma faixa de portas, tanto para os protocolos **TCP** como para **UDP**:

```
<?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 port="1025-65535" protocol="tcp"/>
  <port port="1025-65535" protocol="udp"/>
</zone>
```

Para alterar as configurações dessa zona, adicionar ou remover seções para adicionar portos, encaminhar portos, serviços, e assim por diante.

Recursos adicionais

- Para mais informações, consulte as páginas do manual **firewalld.zone**.

5.10.9. Utilização de metas de zona para definir o comportamento padrão para o tráfego de entrada

Para cada zona, você pode definir um comportamento padrão que trata do tráfego de entrada que não é especificado. Tal comportamento é definido pela definição do objetivo da zona. Há quatro opções - **default**, **ACCEPT**, **REJECT**, e **DROP**. Ao definir o alvo para **ACCEPT**, você aceita todos os pacotes de entrada, exceto aqueles desabilitados por uma regra específica. Se você definir a meta para **REJECT** ou **DROP**, você desabilita todos os pacotes de entrada, exceto aqueles que você permitiu em regras específicas. Quando os pacotes são rejeitados, a máquina de origem é informada sobre a rejeição, enquanto não há informação enviada quando os pacotes são descartados.

Procedimento

Estabelecer uma meta para uma zona:

1. Liste as informações para a zona específica para ver o alvo padrão:

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

2. Estabelecer uma nova meta na zona:

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

5.11. UTILIZAÇÃO DE ZONAS PARA GERENCIAR O TRÁFEGO DE ENTRADA, DEPENDENDO DE UMA FONTE

5.11.1. Utilização de zonas para gerenciar o tráfego de entrada, dependendo de uma fonte

Você pode usar zonas para gerenciar o tráfego de entrada com base em sua fonte. Isso permite classificar o tráfego de entrada e encaminhá-lo através de diferentes zonas para permitir ou não serviços que podem ser alcançados por esse tráfego.

Se você acrescentar uma fonte a uma zona, a zona se torna ativa e qualquer tráfego de entrada dessa fonte será direcionado através dela. Você pode especificar configurações diferentes para cada zona, que são aplicadas ao tráfego de acordo com as fontes dadas. Você pode usar mais zonas mesmo que você tenha apenas uma interface de rede.

5.11.2. Adicionando uma fonte

Para encaminhar o tráfego de entrada para uma fonte específica, acrescente a fonte a essa zona. A fonte pode ser um endereço IP ou uma máscara IP na notação Classless Inter-domain Routing (CIDR).

NOTA

Caso você acrescente múltiplas zonas com uma faixa de rede sobreposta, elas são ordenadas alfanumericamente pelo nome da zona e somente a primeira é considerada.

- Para definir a fonte na zona atual:

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

- Para definir o endereço IP de origem para uma zona específica:

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

O procedimento a seguir permite todo o tráfego de entrada do site *192.168.2.15* na zona **trusted**:

Procedimento

1. Liste todas as zonas disponíveis:

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

2. Adicione a fonte IP à zona de confiança no modo permanente:

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

3. Faça com que as novas configurações sejam persistentes:

```
# Firewall-cmd - tempo de execução a permanente
```

-

5.11.3. Remoção de uma fonte

A remoção de uma fonte da zona corta o tráfego proveniente da mesma.

Procedimento

1. Liste as fontes permitidas para a zona requerida:

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

2. Remover a fonte da zona permanentemente:

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

3. Faça com que as novas configurações sejam persistentes:

```
# Firewall-cmd - tempo de execução a permanente
```

5.11.4. Adicionando uma porta de origem

Para permitir a ordenação do tráfego com base em um porto de origem, especifique um porto de origem usando a opção **--add-source-port**. Você também pode combinar isto com a opção **--add-source** para limitar o tráfego a um determinado endereço IP ou faixa IP.

Procedimento

1. Para adicionar uma porta de origem:

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

5.11.5. Remoção de uma porta de origem

Ao remover um porto de origem, você desabilita a ordenação do tráfego com base em um porto de origem.

Procedimento

1. Para remover um porto de origem:

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

5.11.6. Usando zonas e fontes para permitir um serviço apenas para um domínio específico

Para permitir que o tráfego de uma rede específica utilize um serviço em uma máquina, utilize zonas e fonte. O seguinte procedimento permite que o tráfego de *192.168.1.0/24* possa alcançar o serviço *HTTP* enquanto qualquer outro tráfego é bloqueado.

Procedimento

1. Liste todas as zonas disponíveis:

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

2. Adicione a fonte à zona de confiança para encaminhar o tráfego proveniente da fonte através da zona:

```
# firewall-cmd --zone=trusted --add-source=192.168.1.0/24
```

3. Adicione o serviço *http* na zona de confiança:

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

4. Faça com que as novas configurações sejam persistentes:

```
# Firewall-cmd - tempo de execução a permanente
```

5. Verifique se a zona de confiança está ativa e se o serviço é permitido nela:

```
# firewall-cmd --zone=trusted --list-all
trusted (active)
target: ACCEPT
sources: 192.168.1.0/24
services: http
```

5.11.7. Configuração do tráfego aceito por uma zona com base em um protocolo

Você pode permitir que o tráfego de entrada seja aceito por uma zona com base em um protocolo. Todo o tráfego utilizando o protocolo especificado é aceito por uma zona, na qual você pode aplicar outras regras e filtragem.

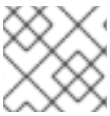
5.11.7.1. Adicionando um protocolo a uma zona

Ao acrescentar um protocolo a uma determinada zona, você permite que todo tráfego com este protocolo seja aceito por esta zona.

Procedimento

1. Para acrescentar um protocolo a uma zona:

```
# firewall-cmd --zone=zone-name --add-protocol=port-name/tcp|udp|sctp|dccp|igmp
```



NOTA

Para receber tráfego multicast, use o valor **igmp** com a opção **--add-protocol**.

5.11.7.2. Remoção de um protocolo de uma zona

Ao remover um protocolo de uma determinada zona, você deixa de aceitar todo o tráfego com base neste protocolo pela zona.

Procedimento

1. Para remover um protocolo de uma zona:

```
# firewall-cmd --zone=zone-name --remove-protocol=port-name/tcp|udp|sctp|dccp|igmp
```

5.12. CONFIGURAÇÃO DE MASCARAMENTO DE ENDEREÇOS IP

O procedimento a seguir descreve como habilitar o mascaramento de IP em seu sistema. O mascaramento de IP esconde máquinas individuais atrás de um gateway quando se acessa a Internet.

Procedimento

1. Para verificar se o mascaramento IP está habilitado (por exemplo, para a zona **external**), digite o seguinte comando como **root**:

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

   O comando imprime **yes** com status de saída **0** se habilitado. Ele imprime **no** com status de saída **1** caso contrário. Se **zone** for omitido, será usada a zona padrão.

2. Para ativar o mascaramento de IP, digite o seguinte comando como **root**:

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

3. Para tornar esta configuração persistente, repita o comando adicionando a opção **--permanent**.

Para desativar o mascaramento de IP, digite o seguinte comando como **root**:

```
# firewall-cmd --zone=externo --remove-masquerade --permanente
```

5.13. ENCAMINHAMENTO DE PORTAS

A redirecionamento de portas usando este método só funciona para tráfego baseado em IPv4. Para a configuração do redirecionamento IPv6, é preciso usar regras ricas.

Para redirecionar para um sistema externo, é necessário permitir o mascaramento. Para mais informações, consulte Configuração de mascaramento de endereços IP .

5.13.1. Adicionando uma porta para redirecionar

Usando **firewalld**, você pode configurar o redirecionamento de portos para que qualquer tráfego de entrada que chegue a um determinado porto em seu sistema seja entregue a outro porto interno de sua escolha ou a um porto externo em outra máquina.

Pré-requisitos

- Antes de redirecionar o tráfego de um porto para outro porto, ou outro endereço, você tem que saber três coisas: qual porta os pacotes chegam, qual protocolo é usado e onde você quer redirecioná-los.

Procedimento

Para redirecionar um porto para outro porto:

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

Para redirecionar uma porta para outra porta em um endereço IP diferente:

1. Acrescentar o porto a ser encaminhado:

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

2. Habilitar o mascaramento:

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

5.13.2. Redirecionando a porta TCP 80 para a porta 88 na mesma máquina

Siga os passos para redirecionar a porta TCP 80 para a porta 88.

Procedimento

1. Redirecionar a porta 80 para a porta 88 para tráfego TCP:

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

2. Faça com que as novas configurações sejam persistentes:

```
# Firewall-cmd - tempo de execução a permanente
```

3. Verifique se o porto está redirecionado:

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

5.13.3. Remoção de um porto redirecionado

Para remover um porto redirecionado:

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

Para remover um porto redirecionado para um endereço diferente, use o seguinte procedimento.

Procedimento

1. Retirar o porto encaminhado:

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

2. Desativar mascarada:

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

5.13.4. Remoção da porta TCP 80 encaminhada para a porta 88 na mesma máquina

Para remover o redirecionamento do porto:

Procedimento

1. Listar portos redirecionados:

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

2. Remover a porta redirecionada do firewall::

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

3. Faça com que as novas configurações sejam persistentes:

```
~]# firewall-cmd --operabilidade a permanente
```

5.14. GERENCIAMENTO DE SOLICITAÇÕES DO ICMP

O **Internet Control Message Protocol** (**ICMP**) é um protocolo de suporte que é usado por vários dispositivos de rede para enviar mensagens de erro e informações operacionais indicando um problema de conexão, por exemplo, que um serviço solicitado não está disponível. **ICMP** difere dos protocolos de transporte como TCP e UDP porque não é usado para trocar dados entre sistemas.

Infelizmente, é possível usar as mensagens **ICMP**, especialmente **echo-request** e **echo-reply**, para revelar informações sobre sua rede e usar indevidamente tais informações para vários tipos de atividades fraudulentas. Portanto, **firewalld** permite bloquear as solicitações **ICMP** para proteger as informações de sua rede.

5.14.1. Listagem e bloqueio de pedidos do ICMP

Listagem ICMP solicitações

As solicitações **ICMP** estão descritas em arquivos XML individuais que estão localizados no diretório **/usr/lib/firewalld/icmptypes/**. Você pode ler estes arquivos para ver uma descrição da solicitação. O comando **firewall-cmd** controla a manipulação das solicitações **ICMP**.

- Para listar todos os tipos disponíveis em **ICMP**:

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

- A solicitação **ICMP** pode ser usada por IPv4, IPv6 ou por ambos os protocolos. Para ver para qual protocolo a solicitação **ICMP** é utilizada:

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

- O status de uma solicitação **ICMP** mostra **yes** se a solicitação estiver atualmente bloqueada ou **no** se não estiver. Para ver se uma solicitação **ICMP** está bloqueada no momento:

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

Bloqueio ou desbloqueio ICMP solicitações

Quando seu servidor bloqueia solicitações do **ICMP**, ele não fornece as informações que normalmente forneceria. No entanto, isso não significa que nenhuma informação seja dada. Os clientes recebem informações de que o pedido específico **ICMP** está sendo bloqueado (rejeitado). O bloqueio das solicitações **ICMP** deve ser considerado cuidadosamente, pois pode causar problemas de comunicação, especialmente com o tráfego IPv6.

- Para ver se uma solicitação **ICMP** está atualmente bloqueada:

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

- Para bloquear um pedido em **ICMP**:

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

- Para remover o bloco para um pedido em **ICMP**:

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

Bloqueio de solicitações ICMP sem fornecer qualquer tipo de informação

Normalmente, se você bloquear solicitações do **ICMP**, os clientes sabem que você está bloqueando. Portanto, um potencial atacante que está farejando endereços IP ao vivo ainda é capaz de ver que seu endereço IP está online. Para esconder completamente estas informações, você tem que descartar todas as solicitações **ICMP**.

- Para bloquear e abandonar todas as solicitações **ICMP**:

1. Defina a meta de sua zona para **DROP**:

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

Agora, todo o tráfego, incluindo os pedidos de **ICMP**, é descartado, exceto o tráfego que você permitiu explicitamente.

- Para bloquear e abandonar certas solicitações **ICMP** e permitir outras:

1. Defina a meta de sua zona para **DROP**:

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

2. Adicionar a inversão de bloco ICMP para bloquear todas as solicitações **ICMP** de uma só vez:

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

3. Adicione o bloco ICMP para aqueles pedidos do site **ICMP** que você deseja permitir:

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

4. Faça com que as novas configurações sejam persistentes:

```
# Firewall-cmd - tempo de execução a permanente
```

O *block inversion* inverte a configuração dos bloqueios de solicitações **ICMP**, de modo que todas as solicitações, que não foram bloqueadas anteriormente, são bloqueadas por causa do alvo de suas mudanças de zona para **DROP**. As solicitações que foram bloqueadas não são bloqueadas. Isto significa que se você deseja desbloquear uma solicitação, deve usar o comando de bloqueio.

- Para reverter a inversão de bloco para um ajuste totalmente permissivo:

1. Defina a meta de sua zona para **default** ou **ACCEPT**:

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

2. Remover todos os blocos adicionados para pedidos em **ICMP**:

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

3. Remova a inversão de bloco **ICMP**:

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

4. Faça com que as novas configurações sejam persistentes:

```
# Firewall-cmd - tempo de execução a permanente
```

5.14.2. Configuração do filtro ICMP usando o GUI

- Para ativar ou desativar um filtro **ICMP**, inicie o **firewall-config** e selecionar a zona de rede cujas mensagens devem ser filtradas. Selecione a aba **ICMP Filter** e selecione a caixa de seleção para cada tipo de mensagem **ICMP** que você deseja filtrar. Desmarque a caixa de seleção para desativar um filtro. Esta configuração é por direção e o padrão permite tudo.

- Para editar um tipo **ICMP**, inicie o **firewall-config** e selecione o modo **Permanent** a partir do menu etiquetado **Configuration**. Ícones adicionais aparecem na parte inferior da janela **Serviços**. Selecione **Sim** no diálogo seguinte para habilitar o mascaramento e para fazer o encaminhamento para outra máquina funcionando.

- Para ativar a inversão do **ICMP Filter**, clique na caixa de seleção **Invert Filter**, à direita. Somente os tipos marcados com **ICMP** são agora aceitos, todos os outros são rejeitados. Em uma zona utilizando o alvo DROP, eles são descartados.

5.15. CONFIGURAÇÃO E CONTROLE DE CONJUNTOS IP USANDO FIREWALLD

Para ver a lista de tipos de conjunto IP suportados por **firewalld**, digite o seguinte comando como root.

```
~]# 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
```

5.15.1. Configuração das opções do conjunto IP usando CLI

Os conjuntos IP podem ser usados nas zonas **firewalld** como fontes e também como fontes em regras ricas. No Red Hat Enterprise Linux, o método preferido é usar os conjuntos de IPs criados com **firewalld** em uma regra direta.

- Para listar os conjuntos de IPs conhecidos por **firewalld** no ambiente permanente, use o seguinte comando como **root**:

```
# firewall-cmd --permanente --get-ipsets
```

- Para adicionar um novo conjunto IP, use o seguinte comando usando o ambiente permanente como **root**:

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

O comando anterior cria um novo conjunto IP com o nome *test* e o tipo **hash:net** para **IPv4**. Para criar um conjunto de IP para uso com **IPv6**, adicione a opção **--option=family=inet6**. Para tornar o novo ajuste efetivo no ambiente de tempo de execução, recarregue **firewalld**.

- Liste o novo conjunto IP com o seguinte comando: **root**:

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

- Para obter mais informações sobre o conjunto IP, use o seguinte comando como **root**:

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

Observe que o conjunto IP não tem nenhuma entrada no momento.

- Para adicionar uma entrada ao conjunto IP *test*, use o seguinte comando como **root**:

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

O comando anterior adiciona o endereço IP *192.168.0.1* ao conjunto IP.

- Para obter a lista de entradas atuais no conjunto IP, use o seguinte comando como **root**:

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

- Gerar um arquivo contendo uma lista de endereços IP, por exemplo:

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

O arquivo com a lista de endereços IP para um conjunto IP deve conter uma entrada por linha. Linhas começando com um hash, um ponto e vírgula, ou linhas vazias são ignoradas.

- Para adicionar os endereços do arquivo *iplist.txt*, use o seguinte comando como **root**:

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

- Para ver a lista ampliada de entradas do conjunto IP, use o seguinte comando como **root**:

```
# 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
```

- Para remover os endereços do conjunto IP e verificar a lista de entradas atualizada, use os seguintes comandos como **root**:

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

- Você pode adicionar o conjunto IP como fonte a uma zona para lidar com todo o tráfego vindo de qualquer um dos endereços listados no conjunto IP com uma zona. Por exemplo, para adicionar o conjunto de IP *test* como fonte à zona *drop* para descartar todos os pacotes vindos de todas as entradas listadas no conjunto de IP *test*, use o seguinte comando como **root**:

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

O prefixo **ipset:** na fonte mostra **firewalld** que a fonte é um conjunto IP e não um endereço IP ou uma faixa de endereços.

Apenas a criação e remoção de conjuntos IP é limitada ao ambiente permanente, todas as outras opções de conjuntos IP podem ser usadas também no ambiente de tempo de execução sem a opção **--permanent**.

ATENÇÃO

A Red Hat não recomenda o uso de conjuntos IP que não são gerenciados através de **firewalld**. Para utilizar tais conjuntos IP, é necessária uma regra direta permanente para referenciar o conjunto, e um serviço personalizado deve ser adicionado para criar estes conjuntos IP. Este serviço precisa ser iniciado antes do firewalld iniciar, caso contrário **firewalld** não é capaz de adicionar as regras diretas usando estes conjuntos. Você pode adicionar as regras diretas permanentes com o arquivo **/etc/firewalld/direct.xml**.

5.16. PRIORIZANDO REGRAS RICAS

Por padrão, as regras ricas são organizadas com base em sua ação de regras. Por exemplo, as regras **deny** têm precedência sobre as regras **allow**. O parâmetro **priority** nas regras ricas fornece aos administradores um controle granulado fino sobre as regras ricas e sua ordem de execução.

5.16.1. Como o parâmetro prioritário organiza as regras em diferentes cadeias

Você pode definir o parâmetro **priority** em uma regra rica para qualquer número entre **-32768** e **32767**, e valores mais baixos têm maior precedência.

O serviço **firewalld** organiza regras com base em seu valor prioritário em diferentes cadeias:

- Prioridade inferior a 0: a regra é redirecionada para uma corrente com o sufixo **_pre**.
- Prioridade maior que 0: a regra é redirecionada para uma cadeia com o sufixo **_post**.
- Prioridade igual a 0: com base na ação, a regra é redirecionada para uma cadeia com o **_log**, **_deny**, ou **_allow** a ação.

Dentro destas subdivisões, **firewalld** ordena as regras com base em seu valor prioritário.

5.16.2. Estabelecendo a prioridade de uma regra rica

O procedimento descreve um exemplo de como criar uma regra rica que usa o parâmetro **priority** para registrar todo o tráfego que não é permitido ou negado por outras regras. Você pode usar esta regra para sinalizar tráfego inesperado.

Procedimento

1. Acrescente uma regra rica com uma precedência muito baixa para registrar todo o tráfego que não tenha sido igualado por outras regras:

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

   O comando limita adicionalmente o número de entradas de registro a **5** por minuto.

2. Opcionalmente, exibir a regra **nftables** que o comando na etapa anterior criou:

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

5.17. CONFIGURAÇÃO DO BLOQUEIO DO FIREWALL

Aplicações ou serviços locais são capazes de alterar a configuração do firewall se estiverem rodando como **root** (por exemplo, **libvirt**). Com este recurso, o administrador pode bloquear a configuração do firewall para que nenhuma aplicação ou apenas as aplicações que são adicionadas à lista de bloqueio

permitam solicitar mudanças no firewall. As configurações de bloqueio padrão são desabilitadas. Se ativada, o usuário pode ter certeza de que não há alterações indesejadas na configuração do firewall feitas por aplicações ou serviços locais.

5.17.1. Configuração de bloqueio usando CLI

- Para consultar se o bloqueio está ativado, use o seguinte comando como **root**:

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

  O comando imprime **yes** com status de saída **0** se o bloqueio estiver ativado. Ele imprime **no** com status de saída **1** caso contrário.

- Para ativar o bloqueio, digite o seguinte comando como **root**:

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

- Para desativar o bloqueio, use o seguinte comando como **root**:

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

5.17.2. Configuração das opções de listas de bloqueio usando CLI

A lista de permissão de bloqueio pode conter comandos, contextos de segurança, usuários e IDs de usuários. Se uma entrada de comando na lista de permissões terminar com um asterisco "*", então todas as linhas de comando que começam com esse comando serão iguais. Se o {\i1}"*" não estiver lá, então o comando absoluto, incluindo os argumentos, deve coincidir.

- O contexto é o contexto de segurança (SELinux) de uma aplicação ou serviço em execução. Para obter o contexto de uma aplicação em execução, use o seguinte comando:

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

  Esse comando retorna todas as aplicações em execução. Encaneie a saída através do **grep** ferramenta para obter a aplicação de interesse. Por exemplo:

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

- Para listar todas as linhas de comando que estão na lista de permissão, digite o seguinte comando como **root**:

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

- Para adicionar um comando *command* à lista de permissão, digite o seguinte comando como **root**:

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

- Para remover um comando *command* da lista de permissão, digite o seguinte comando como **root**:

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

- Para saber se o comando *command* está na lista de permissão, digite o seguinte comando como **root**:

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

O comando imprime **yes** com status de saída **0** se for verdade. Ele imprime **no** com status de saída **1** caso contrário.

- Para listar todos os contextos de segurança que estão na lista de permissão, digite o seguinte comando como **root**:

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

- Para adicionar um contexto *context* à lista de permissão, digite o seguinte comando como **root**:

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

- Para remover um contexto *context* da lista de permissão, digite o seguinte comando como **root**:

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

- Para saber se o contexto *context* está na lista de permissão, digite o seguinte comando como **root**:

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

Imprime **yes** com status de saída **0**, se verdadeiro, imprime **no** com status de saída **1** caso contrário.

- Para listar todos os IDs de usuário que estão na lista de permissão, digite o seguinte comando como **root**:

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

- Para adicionar um ID de usuário *uid* à lista de permissões, digite o seguinte comando como **root**:

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

- Para remover um ID de usuário *uid* da lista de permissão, digite o seguinte comando como **root**:

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

- Para consultar se o ID do usuário *uid* está na lista de permissão, digite o seguinte comando:

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

Imprime **yes** com status de saída **0**, se verdadeiro, imprime **no** com status de saída **1** caso contrário.

- Para listar todos os nomes de usuários que estão na lista de permissão, digite o seguinte comando como **root**:

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

- Para adicionar um nome de usuário *user* à lista de permissão, digite o seguinte comando como **root**:

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

- Para remover um nome de usuário *user* da lista de permissão, digite o seguinte comando como **root**:

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

- Para consultar se o nome do usuário *user* está na lista de permissão, digite o seguinte comando:

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

Imprime **yes** com status de saída **0**, se verdadeiro, imprime **no** com status de saída **1** caso contrário.

5.17.3. Configuração de opções de lista de bloqueio usando arquivos de configuração

O arquivo de configuração padrão da lista de permissão contém o contexto **NetworkManager** e o contexto padrão de **libvirt**. O ID de usuário 0 também está na lista.

```
<?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>
```

A seguir, um exemplo de arquivo de configuração de lista de permissão que permite todos os comandos para o utilitário **firewall-cmd**, para um usuário chamado *user* cujo ID de usuário é **815**:

```
<?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>
```

Este exemplo mostra tanto **user id** como **user name**, mas apenas uma opção é necessária. Python é o intérprete e está preparado para a linha de comando. Você também pode usar um comando específico, por exemplo:

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

Nesse exemplo, somente o comando **--lockdown-on** é permitido.

No Red Hat Enterprise Linux, todos os utilitários são colocados no diretório **/usr/bin/** e o diretório **/bin/** está vinculado simbolicamente ao diretório **/usr/bin/**. Em outras palavras, embora o caminho para **firewall-cmd** quando inserido como **root** possa ser resolvido para **/bin/firewall-cmd**, **/usr/bin/firewall-cmd** pode agora ser usado. Todos os novos scripts devem usar o novo local. Mas esteja ciente de que se os scripts que rodam como **root** forem escritos para usar o caminho **/bin/firewall-cmd**, então esse caminho de comando deve ser adicionado na lista de permissão, além do caminho **/usr/bin/firewall-cmd** tradicionalmente usado apenas para usuários não**root**.

O * no final do atributo do nome de um comando significa que todos os comandos que começam com esta string correspondem. Se o * não estiver lá, então o comando absoluto, incluindo argumentos, deve coincidir.

5.18. LOG PARA PACOTES NEGADOS

Com a opção **LogDenied** no site **firewalld**, é possível adicionar um mecanismo simples de registro para pacotes negados. Estes são os pacotes que são rejeitados ou descartados. Para alterar a configuração do registro, edite o arquivo **/etc/firewalld/firewalld.conf** ou use a linha de comando ou a ferramenta de configuração GUI.

Se **LogDenied** estiver habilitado, as regras de registro são adicionadas logo antes das regras de rejeição e desistência nas cadeias INPUT, FORWARD e OUTPUT para as regras padrão e também as regras finais de rejeição e desistência nas zonas. Os valores possíveis para esta configuração são: **all** , **unicast**, **broadcast**, **multicast**, e **off**. A configuração padrão é **off**. Com a configuração **unicast**, **broadcast**, e **multicast**, a correspondência **pkttype** é usada para combinar com o tipo de pacote de camada de link. Com **all**, todos os pacotes são registrados.

Para listar a configuração real **LogDenied** com firewall-cmd, use o seguinte comando como **root**:

```
# firewall-cmd --get-log-denied
off
```

Para alterar a configuração **LogDenied**, use o seguinte comando como **root**:

```
# firewall-cmd --set-log-denied=all
success
```

Para alterar a configuração **LogDenied** com a ferramenta de configuração **firewalld** GUI, inicie **firewall-config**clique no menu **Options** e selecione **Change Log Denied**. A janela **LogDenied** aparece. Selecione a nova configuração **LogDenied** no menu e clique em OK.

5.19. INFORMAÇÕES RELACIONADAS

As seguintes fontes de informação fornecem recursos adicionais em relação a **firewalld**.

Documentação instalada

- **firewalld(1)** página man - descreve opções de comando para **firewalld**.
- **firewalld.conf(5)** página man - contém informações para configurar **firewalld**.
- **firewall-cmd(1)** man page - descreve opções de comando para o cliente de linha de comando **firewalld**.
- **firewall-config(1)** man page - descreve as configurações para o **firewall-config** ferramenta.

- **firewall-offline-cmd(1)** man page - descreve as opções de comando para o cliente de linha de comando offline **firewalld**.
- **firewalld.icmptype(5)** man page - descreve arquivos de configuração XML para **ICMP** filtragem.
- **firewalld.ipset(5)** man page - descreve arquivos de configuração XML para os conjuntos **firewalld IP** .
- **firewalld.service(5)** man page - descreve arquivos de configuração XML para **firewalld service**.
- **firewalld.zone(5)** man page - descreve os arquivos de configuração XML para a configuração da zona **firewalld**.
- **firewalld.direct(5)** página man - descreve o arquivo de configuração da interface direta **firewalld**.
- **firewalld.lockdown-whitelist(5)** man page - descreve o arquivo de configuração da lista **firewalld** lockdown allowlist.
- **firewalld.richlanguage(5)** man page - descreve a sintaxe da regra da linguagem rica **firewalld**.
- **firewalld.zones(5)** man page - descrição geral do que são zonas e como configurá-las.
- **firewalld.dbus(5)** página man - descreve a interface **D-Bus** de **firewalld**.

Documentação on-line

- http://www.firewalld.org/ - **firewalld** página inicial.

CAPÍTULO 6. COMEÇANDO COM NFTABLES

A estrutura **nftables** oferece facilidades de classificação de pacotes e é o sucessor designado para as ferramentas **iptables**, **ip6tables**, **arptables**, e **ebtables**. Ela oferece inúmeras melhorias em conveniência, características e desempenho em relação às ferramentas de filtragem de pacotes anteriores, mais notadamente:

- tabelas de pesquisa em vez de processamento linear
- uma estrutura única para ambos os protocolos **IPv4** e **IPv6**
- regras todas aplicadas atomicamente em vez de buscar, atualizar e armazenar um conjunto completo de regras
- suporte para depuração e rastreamento no conjunto de regras (**nftrace**) e monitoramento de eventos de rastreamento (na ferramenta **nft**)
- sintaxe mais consistente e compacta, sem extensões específicas de protocolo
- uma API Netlink para aplicações de terceiros

Da mesma forma que **iptables**, **nftables** utiliza tabelas para o armazenamento de correntes. As cadeias contêm regras individuais para a realização de ações. A ferramenta **nft** substitui todas as ferramentas das estruturas anteriores de filtragem de pacotes. A biblioteca **libnftnl** pode ser usada para interação de baixo nível com **nftables** Netlink API sobre a biblioteca **libmnl**.

O efeito dos módulos sobre o conjunto de regras **nftables** pode ser observado usando o comando **nft** list rule set. Como estas ferramentas adicionam tabelas, correntes, regras, conjuntos e outros objetos ao conjunto de regras **nftables**, esteja ciente de que **nftables** operações do conjunto de regras, como o comando **nft flush ruleset**, podem afetar os conjuntos de regras instalados usando os comandos herdados anteriormente separados.

6.1. MIGRANDO DE IPTABLES PARA NFTABLES

Se você atualizou seu servidor para o RHEL 8 ou sua configuração de firewall ainda usa as regras **iptables**, você pode migrar suas regras **iptables** para **nftables**.

6.1.1. Quando usar firewalld, nftables, ou iptables

A seguir, uma breve visão geral em que cenário você deve utilizar uma das seguintes utilidades:

- **firewalld**: Use o utilitário **firewalld** para casos simples de uso de firewall. O utilitário é fácil de usar e cobre os casos de uso típico para estes cenários.
- **nftables**: Use o utilitário **nftables** para criar firewalls complexos e de desempenho crítico, como para toda uma rede.
- **iptables**: O utilitário **iptables** no Red Hat Enterprise Linux 8 usa a API do kernel **nf_tables** ao invés do back end **legacy**. A API **nf_tables** fornece compatibilidade retroativa para que scripts que usam os comandos **iptables** ainda funcionem no Red Hat Enterprise Linux 8. Para novos scripts de firewall, a Red Hat recomenda usar **nftables**.

IMPORTANTE

Para evitar que os diferentes serviços de firewall influenciem uns aos outros, execute apenas um deles em um host RHEL, e desabilite os outros serviços.

6.1.2. Conversão de regras iptables em regras nftables

O Red Hat Enterprise Linux 8 fornece as ferramentas **iptables-translate** e **ip6tables-translate** para converter as regras existentes **iptables** ou **ip6tables** em regras equivalentes para **nftables**.

Observe que algumas extensões carecem de suporte de tradução. Se tal extensão existir, a ferramenta imprime a regra não traduzida prefixada com o sinal **#**. Por exemplo:

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

Além disso, os usuários podem usar as ferramentas **iptables-restore-translate** e **ip6tables-restore-translate** para traduzir um lixão de regras. Note que antes disso, os usuários podem usar os comandos **iptables-save** ou **ip6tables-save** para imprimir um dump das regras atuais. Por exemplo:

```
# iptables-save >/tmp/iptables.dump
# iptables-restore-translate -f /tmp/iptables.dump

# Translated by iptables-restore-translate v1.8.0 on Wed Oct 17 17:00:13 2018
add table ip nat
...
```

Para mais informações e uma lista de opções e valores possíveis, digite o comando **iptables-translate --help**.

6.2. ESCREVER E EXECUTAR SCRIPTS NFTABLES

A estrutura **nftables** fornece um ambiente de script nativo que traz um grande benefício sobre o uso de scripts shell para manter as regras de firewall: a execução de scripts é atômica. Isto significa que o sistema ou aplica o script inteiro ou impede a execução se ocorrer um erro. Isto garante que o firewall esteja sempre em um estado consistente.

Além disso, o ambiente de script **nftables** permite que os administradores o façam:

- adicionar comentários
- definir variáveis
- incluir outros arquivos do conjunto de regras

Esta seção explica como utilizar estes recursos, assim como a criação e execução de scripts **nftables**.

Quando você instala o pacote **nftables**, o Red Hat Enterprise Linux cria automaticamente ***.nft** scripts no diretório **/etc/nftables/**. Estes scripts contêm comandos que criam tabelas e cadeias vazias para diferentes propósitos. Você pode estender estes arquivos ou escrever seus scripts.

6.2.1. O cabeçalho do script necessário em nftables script

Semelhante a outros scripts, **nftables** scripts requerem uma seqüência de shebang na primeira linha do script que define a diretiva do intérprete.

Um script **nftables** deve sempre começar com a seguinte linha:

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

IMPORTANTE

Se você omitir o parâmetro **-f**, o utilitário **nft** não lê o script e exibe **Error: syntax error, unexpected newline, expecting string**.

6.2.2. Formatos de scripts nftables suportados

O ambiente **nftables** suporta scripts nos seguintes formatos:

- Você pode escrever um script no mesmo formato que o comando **nft list ruleset** exibe o conjunto de regras:

```
#!/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
  }
}
```

- Você pode usar a mesma sintaxe para comandos como em **nft** comandos:

```
#!/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
```

6.2.3. Executando nftables scripts

Para executar um script **nftables**, o script deve ser executável. Somente se o script for incluído em outro script, ele não precisa ser executável. O procedimento descreve como tornar um script executável e executar o script.

Pré-requisitos

- O procedimento desta seção pressupõe que você tenha armazenado um script **nftables** no arquivo **/etc/nftables/example_firewall.nft**.

Procedimento

1. Passos que são necessários apenas uma vez:
   a. Opcionalmente, defina o dono do roteiro para **root**:

```
# raiz de enxada /etc/nftables/example_firewall.nft
```

   b. Tornar o roteiro executável para o proprietário:

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

2. Execute o roteiro:

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

   Se nenhuma saída for exibida, o sistema executou o script com sucesso.

IMPORTANTE

Mesmo que **nft** execute o script com sucesso, regras colocadas incorretamente, parâmetros ausentes ou outros problemas no script podem fazer com que o firewall não se comporte como esperado.

Recursos adicionais

- Para detalhes sobre como definir o proprietário de um arquivo, consulte a página de manual **chown(1)**.
- Para detalhes sobre a definição de permissões de um arquivo, consulte a página de manual **chmod(1)**.
- Seção 6.2.7, "Carregamento automático das regras nftables quando o sistema inicia"

6.2.4. Usando comentários em scripts nftables

O ambiente **nftables** interpreta tudo à direita de um personagem **#** como um comentário.

Exemplo 6.1. Comentários em um roteiro nftables

Os comentários podem começar no início de uma linha, assim como ao lado de um comando:

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

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

–

6.2.5. Usando variáveis em um script nftables

Para definir uma variável em um script **nftables**, use a palavra-chave **define**. Você pode armazenar valores individuais e conjuntos anônimos em uma variável. Para cenários mais complexos, use conjuntos ou mapas de veredictos.

Variáveis com um único valor
O exemplo a seguir define uma variável chamada **INET_DEV** com o valor **enp1s0**:

```
define INET_DEV = enp1s0
```

Você pode usar a variável no script escrevendo o sinal **$** seguido do nome da variável:

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

Variáveis que contêm um conjunto anônimo
O exemplo a seguir define uma variável que contém um conjunto anônimo:

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

Você pode usar a variável no script escrevendo o sinal **$** seguido do nome da variável:

```
adicionar exemplo de regra inet exemplo de tabela_chain ip daddr $DNS_SERVERS aceitar
```

NOTA

Observe que os suportes encaracolados têm uma semântica especial quando você os usa em uma regra, pois indicam que a variável representa um conjunto.

Recursos adicionais

- Para detalhes sobre os conjuntos, ver Seção 6.5, “Usando conjuntos em comandos nftables” .
- Para detalhes sobre os mapas de veredictos, veja Seção 6.6, “Usando mapas de veredictos em comandos nftables”.

6.2.6. Incluindo arquivos em um script nftables

O ambiente **nftables** permite que os administradores incluam outros scripts usando a declaração **include**.

Se você especificar apenas um nome de arquivo sem um caminho absoluto ou relativo, **nftables** inclui arquivos do caminho de busca padrão, que está definido para **/etc** no Red Hat Enterprise Linux.

Exemplo 6.2. Incluindo arquivos do diretório de busca padrão

Para incluir um arquivo do diretório de busca padrão:

```
incluem "exemplo.nft"
```

Exemplo 6.3. Incluindo todos os arquivos *.nft de um diretório

Para incluir todos os arquivos que terminam em ***.nft** que estão armazenados no diretório **/etc/nftables/rulesets/**:

```
incluem "/etc/nftables/rulesets/*.nft\"
```

Observe que a declaração **include** não corresponde a arquivos que começam com um ponto.

Recursos adicionais

- Para mais detalhes, consulte a seção **Include files** na página de manual **nft(8)**.

6.2.7. Carregamento automático das regras nftables quando o sistema inicia

O serviço **nftables** systemd carrega scripts de firewall que estão incluídos no arquivo **/etc/sysconfig/nftables.conf**. Esta seção explica como carregar as regras de firewall quando o sistema inicia.

Pré-requisitos

- Os scripts **nftables** são armazenados no diretório **/etc/nftables/**.

Procedimento

1. Edite o arquivo **/etc/sysconfig/nftables.conf**.
   - Se você melhorar ***.nft** scripts criados em **/etc/nftables/** ao instalar o pacote **nftables**, descomente a declaração **include** para estes scripts.
   - Se você escrever scripts a partir do zero, adicione declarações em **include** para incluir estes scripts. Por exemplo, para carregar o **/etc/nftables/*example*.nft** quando o serviço **nftables** for iniciado, acrescente:

     ```
     incluem "/etc/nftables/example.nft\"
     ```

2. Habilite o serviço **nftables**.

   ```
   # systemctl habilita nftables
   ```

3. Opcionalmente, inicie o serviço **nftables** para carregar as regras de firewall sem reiniciar o sistema:

   ```
   # systemctl start nftables
   ```

Recursos adicionais

- Seção 6.2.2, “Formatos de scripts nftables suportados”

6.3. CRIAÇÃO E GERENCIAMENTO DE TABELAS, CORRENTES E REGRAS NFTABLES

Esta seção explica como exibir os conjuntos de regras **nftables**, e como gerenciá-los.

6.3.1. Valores padrão de prioridade da cadeia e nomes textuais

Quando você cria uma cadeia, o **priority** pode definir um valor inteiro ou um nome padrão que especifica a ordem na qual as cadeias com o mesmo valor **hook** atravessam.

Os nomes e valores são definidos com base em quais prioridades são utilizados pelo **xtables** ao registrar suas cadeias padrão.

NOTA

O comando **nft list chains** exibe valores de prioridade textual por padrão. Você pode visualizar o valor numérico passando a opção **-y** para o comando.

Exemplo 6.4. Usando um valor textual para definir a prioridade

O seguinte comando cria uma cadeia chamada **example_chain** em **example_table** usando o valor de prioridade padrão **50**:

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

Como a prioridade é um valor padrão, você pode, alternativamente, usar o valor textual:

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

Tabela 6.1. Nomes de prioridade padrão, família e matriz de compatibilidade de ganchos

Nome	Valor	Famílias	Anzóis
raw	-300	**ip**, **ip6**, **inet**	todos
mangle	-150	**ip**, **ip6**, **inet**	todos
dstnat	-100	**ip**, **ip6**, **inet**	pré-encaminhamento
filter	0	**ip**, **ip6**, **inet**, **arp**, **netdev**	todos
security	50	**ip**, **ip6**, **inet**	todos
srcnat	100	**ip**, **ip6**, **inet**	pós-transplante

Todas as famílias utilizam os mesmos valores, mas a família **bridge** utiliza os seguintes valores:

Tabela 6.2. Nomes de prioridade padrão, e compatibilidade de ganchos para a família bridge

Nome	Valor	Anzóis
dstnat	-300	pré-encaminhamento
filter	-200	todos
out	100	saída
srcnat	300	pós-transplante

Recursos adicionais

- Para obter detalhes sobre outras ações que você pode executar em cadeias, consulte a seção **Chains** na página de manual **nft(8)**.

6.3.2. Exibição de conjuntos de regras nftables

Os conjuntos de regras do **nftables** contêm tabelas, correntes e regras. Esta seção explica como exibir esses conjuntos de regras.

Procedimento

- Para exibir todos os conjuntos de regras, entre:

```
# 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
  }
}
```

NOTA

Por padrão, **nftables** não pré-cria tabelas. Como conseqüência, exibindo a regra definida em um host sem nenhuma tabela, o comando **nft list ruleset** não mostra nenhuma saída.

6.3.3. Criando uma tabela nftables

Uma tabela em **nftables** é um espaço de nomes que contém uma coleção de correntes, regras, conjuntos e outros objetos. Esta seção explica como criar uma tabela.

Cada tabela deve ter uma família de endereços definida. A família de endereços de uma tabela define que tipos de endereços a tabela processa. Você pode definir uma das seguintes famílias de endereços ao criar uma tabela:

- **ip**: Combina somente pacotes IPv4. Este é o padrão se você não especificar uma família de endereços.
- **ip6**: Combina apenas pacotes IPv6.

- **inet**: Combina pacotes IPv4 e IPv6.
- **arp**: Corresponde aos pacotes do protocolo de resolução de endereços IPv4 (ARP).
- **bridge**: Combina pacotes que atravessam um dispositivo de ponte.
- **netdev**: Combina pacotes de entrada.

Procedimento

1. Use o comando **nft add table** para criar uma nova tabela. Por exemplo, para criar uma tabela chamada **example_table** que processa pacotes IPv4 e IPv6:

```
# nft adicionar tabela inet exemplo_tabela
```

2. Opcionalmente, liste todas as tabelas do conjunto de regras:

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

Recursos adicionais

- Para mais detalhes sobre famílias de endereços, consulte a seção **Address families** na página de manual **nft(8)**.
- Para detalhes sobre outras ações que você pode executar em tabelas, consulte a seção **Tables** na página de manual **nft(8)**.

6.3.4. Criando uma cadeia nftables

As correntes são recipientes para regras. Existem os dois tipos de regras a seguir:

- Cadeia base: Você pode usar cadeias de base como um ponto de entrada para pacotes da pilha de rede.
- Corrente regular: Você pode usar correntes regulares como um alvo **jump** e para organizar melhor as regras.

O procedimento descreve como adicionar uma cadeia de base a uma tabela existente.

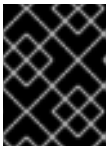
Pré-requisitos

- A tabela à qual se deseja acrescentar a nova cadeia existe.

Procedimento

1. Use o comando **nft add chain** para criar uma nova cadeia. Por exemplo, para criar uma cadeia chamada **example_chain** em **example_table**:

```
# nft add chain inet example_table example_chain { type filter hook input priority 0 {\i1}; policy accept {\i1}
```



IMPORTANTE

Para evitar que a casca interprete os ponto-e-vírgula como o fim do comando, você deve escapar dos pontos-e-vírgula com uma barra invertida.

Esta corrente filtra os pacotes de entrada. O parâmetro **priority** especifica a ordem na qual **nftables** processa cadeias com o mesmo valor de gancho. Um valor de prioridade mais baixo tem precedência sobre os mais altos. O parâmetro **policy** define a ação padrão para as regras nesta cadeia. Observe que se você estiver conectado remotamente ao servidor e definir a política padrão para **drop**, você será desconectado imediatamente se nenhuma outra regra permitir o acesso remoto.

2. Opcionalmente, exibir todas as correntes:

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

Recursos adicionais

- Para mais detalhes sobre famílias de endereços, consulte a seção **Address families** na página de manual **nft(8)**.
- Para obter detalhes sobre outras ações que você pode executar em cadeias, consulte a seção **Chains** na página de manual **nft(8)**.

6.3.5. Adicionando uma regra a uma cadeia de nftables

Esta seção explica como adicionar uma regra a uma cadeia **nftables** existente. Por padrão, o comando **nftables add rule** acrescenta uma nova regra ao final da cadeia.

Se você quiser inserir uma regra no início da cadeia, veja Seção 6.3.6, “Inserindo uma regra em uma cadeia de nftables”.

Pré-requisitos

- A cadeia à qual se deseja acrescentar a regra existe.

Procedimento

1. Para adicionar uma nova regra, use o comando **nft add rule**. Por exemplo, para adicionar uma regra ao **example_chain** no **example_table** que permite o tráfego TCP na porta 22:

```
# nft adicionar regra inet example_table example_chain tcp dport 22 accept
```

Em vez do número da porta, você pode, alternativamente, especificar o nome do serviço. No exemplo, você poderia usar **ssh** em vez do número da porta **22**. Observe que um nome de serviço é resolvido para um número de porta com base em sua entrada no arquivo **/etc/services**.

2. Opcionalmente, exibir todas as correntes e suas regras em **example_table**:

```
-
```

```
# 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
  }
}
```

Recursos adicionais

- Para mais detalhes sobre famílias de endereços, consulte a seção **Address families** na página de manual **nft(8)**.
- Para detalhes sobre outras ações que você pode executar sobre regras, consulte a seção **Rules** na página de manual **nft(8)**.

6.3.6. Inserindo uma regra em uma cadeia de nftables

Esta seção explica como inserir uma regra no início de uma cadeia existente **nftables** usando o comando **nftables insert rule**. Se você quiser, ao invés disso, adicionar uma regra ao final de uma cadeia, veja Seção 6.3.5, “Adicionando uma regra a uma cadeia de nftables” .

Pré-requisitos

- A cadeia à qual se deseja acrescentar a regra existe.

Procedimento

1. Para inserir uma nova regra, use o comando **nft insert rule**. Por exemplo, para inserir uma regra no **example_chain** no **example_table** que permite o tráfego TCP na porta 22:

   ```
   # nft inserir regra inet example_table example_chain tcp dport 22 accept
   ```

   Você pode, alternativamente, especificar o nome do serviço em vez do número da porta. No exemplo, você poderia usar **ssh** ao invés do número da porta **22**. Observe que um nome de serviço é resolvido para um número de porta com base em sua entrada no arquivo **/etc/services**.

2. Opcionalmente, exibir todas as correntes e suas regras em **example_table**:

   ```
   # 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
       ...
     }
   }
   ```

Recursos adicionais

- Para mais detalhes sobre famílias de endereços, consulte a seção **Address families** na página de manual **nft(8)**.

- Para detalhes sobre outras ações que você pode executar sobre regras, consulte a seção **Rules** na página de manual **nft(8)**.

6.4. CONFIGURAÇÃO DE NAT USANDO NFTABLES

Com **nftables**, você pode configurar os seguintes tipos de tradução de endereços de rede (NAT):

- Mascarading
- Fonte NAT (SNAT)
- Destino NAT (DNAT)

6.4.1. Os diferentes tipos de NAT: mascaramento, NAT de origem e NAT de destino

Estes são os diferentes tipos de tradução de endereços de rede (NAT):

Mascaramento e fonte NAT (SNAT)

Use um desses tipos de NAT para alterar o endereço IP de origem dos pacotes. Por exemplo, os provedores de Internet não roteiam faixas IP reservadas, tais como **10.0.0.0**/**8**. Se você utiliza faixas de IP reservadas em sua rede e os usuários devem ser capazes de alcançar servidores na Internet, mapeie o endereço IP de origem dos pacotes a partir dessas faixas para um endereço IP público. Tanto o mascaramento quanto o SNAT são muito semelhantes. As diferenças são:

- O mascaramento utiliza automaticamente o endereço IP da interface de saída. Portanto, use o mascarading se a interface de saída usar um endereço IP dinâmico.
- SNAT define o endereço IP de origem dos pacotes para um IP especificado e não procura dinamicamente o IP da interface de saída. Portanto, o SNAT é mais rápido que o mascaramento. Use o SNAT se a interface de saída usar um endereço IP fixo.

Destino NAT (DNAT)

Use este tipo de NAT para encaminhar o tráfego de entrada para um host diferente. Por exemplo, se seu servidor web usa um endereço IP de uma faixa IP reservada e, portanto, não é diretamente acessível da Internet, você pode definir uma regra DNAT no roteador para redirecionar o tráfego de entrada para este servidor.

6.4.2. Configuração de mascaramento usando nftables

O mascaramento permite que um roteador altere dinamicamente o IP de origem dos pacotes enviados através de uma interface para o endereço IP da interface. Isto significa que se a interface recebe um novo IP atribuído, **nftables** usa automaticamente o novo IP ao substituir o IP de origem.

O procedimento a seguir descreve como substituir o IP de origem dos pacotes que saem do host através da interface **ens3** para o conjunto IP em **ens3**.

Procedimento

1. Criar uma mesa:

```
# nft adicionar tabela nat
```

2. Acrescente as cadeias **prerouting** e **postrouting** à tabela:

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

IMPORTANTE

Mesmo que você não acrescente uma regra à cadeia **prerouting**, a estrutura **nftables** exige que esta cadeia corresponda às respostas dos pacotes recebidos.

Observe que você deve passar a opção **--** para o comando **nft** para evitar que o shell interprete o valor de prioridade negativa como uma opção do comando **nft**.

3. Adicione uma regra à cadeia **postrouting** que combine com os pacotes de saída na interface **ens3**:

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

6.4.3. Configuração da fonte NAT usando nftables

Em um roteador, Source NAT (SNAT) permite alterar o IP dos pacotes enviados através de uma interface para um endereço IP específico.

O procedimento a seguir descreve como substituir o IP de origem dos pacotes que deixam o roteador através da interface **ens3** para **192.0.2.1**.

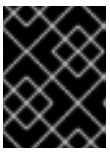
Procedimento

1. Criar uma mesa:

```
# nft adicionar tabela nat
```

2. Acrescente as cadeias **prerouting** e **postrouting** à tabela:

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



IMPORTANTE

Mesmo que você não acrescente uma regra à cadeia **postrouting**, a estrutura **nftables** exige que esta cadeia combine as respostas dos pacotes de saída.

Observe que você deve passar a opção **--** para o comando **nft** para evitar que o shell interprete o valor de prioridade negativa como uma opção do comando **nft**.

3. Adicione uma regra à cadeia **postrouting** que substitui o IP de origem dos pacotes de saída através de **ens3** por **192.0.2.1**:

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

Recursos adicionais

- Seção 6.7.2, "Encaminhamento de pacotes de entrada em uma porta local específica para um host diferente"

6.4.4. Configuração do NAT de destino usando nftables

O NAT de destino permite redirecionar o tráfego em um roteador para um host que não é diretamente acessível a partir da Internet.

O procedimento a seguir descreve como redirecionar o tráfego de entrada enviado para a porta **80** e **443** do roteador para o host com o endereço IP **192.0.2.1**.

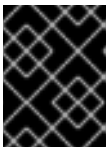
Procedimento

1. Criar uma mesa:

```
# nft adicionar tabela nat
```

2. Acrescente as cadeias **prerouting** e **postrouting** à tabela:

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



IMPORTANTE

Mesmo que você não acrescente uma regra à cadeia **postrouting**, a estrutura **nftables** exige que esta cadeia combine as respostas dos pacotes de saída.

Observe que você deve passar a opção **--** para o comando **nft** para evitar que o shell interprete o valor de prioridade negativa como uma opção do comando **nft**.

3. Adicione uma regra à cadeia **prerouting** que redireciona o tráfego de entrada na interface **ens3** enviada para a porta **80** e **443** para o host com o IP **192.0.2.1**:

```
# nft adicionar regra nat prerouting iifname ens3 tcp dport { 80, 443 } dnat a 192.0.2.1
```

4. Dependendo de seu ambiente, adicione uma regra SNAT ou mascarada para alterar o endereço de origem:

   a. Se a interface **ens3** utilizava endereços IP dinâmicos, acrescente uma regra de mascaramento:

```
# nft adicionar regra nat postrouting oifname {\i1}"ens3} mascarada
```

   b. Se a interface **ens3** usa um endereço IP estático, adicione uma regra SNAT. Por exemplo, se o **ens3** usa o endereço IP **198.51.100.1**:

```
nft adicionar a regra nat postrouting oifname {\i1}"ens3} snat a 198.51.100.1
```

Recursos adicionais

- Seção 6.4.1, “Os diferentes tipos de NAT: mascaramento, NAT de origem e NAT de destino”

6.5. USANDO CONJUNTOS EM COMANDOS NFTABLES

A estrutura **nftables** suporta nativamente conjuntos. Você pode usar conjuntos, por exemplo, se uma regra deve corresponder a múltiplos endereços IP, números de porta, interfaces ou qualquer outro critério de correspondência.

6.5.1. Utilização de conjuntos anônimos em nftables

Um conjunto anônimo contém valores separados por vírgulas entre parênteses, como **{ 22, 80, 443 }**, que você usa diretamente em uma regra. Você também pode usar conjuntos anônimos também para endereços IP ou qualquer outro critério de correspondência.

A desvantagem dos conjuntos anônimos é que, se você quiser mudar o conjunto, você deve substituir a regra. Para uma solução dinâmica, use os conjuntos nomeados como descrito em Seção 6.5.2, "Usando conjuntos nomeados em nftables".

Pré-requisitos

- A cadeia **example_chain** e a tabela **example_table** da família **inet** existe.

Procedimento

1. Por exemplo, para adicionar uma regra a **example_chain** em **example_table** que permite o tráfego de entrada para a porta **22**, **80** e **443**:

```
# nft adicionar regra inet example_table example_chain tcp dport { 22, 80, 443 } aceitar
```

2. Opcionalmente, exibir todas as correntes e suas regras em **example_table**:

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

6.5.2. Usando conjuntos nomeados em nftables

A estrutura **nftables** suporta conjuntos de nomes mutáveis. Um conjunto nomeado é uma lista ou gama de elementos que você pode usar em múltiplas regras dentro de uma tabela. Outro benefício sobre os conjuntos anônimos é que você pode atualizar um conjunto nomeado sem substituir as regras que utilizam o conjunto.

Quando você cria um conjunto nomeado, você deve especificar o tipo de elementos que o conjunto contém. Você pode definir os seguintes tipos:

- **ipv4_addr** para um conjunto que contenha endereços ou faixas IPv4, como **192.0.2.1** ou **192.0.2.0/24**.
- **ipv6_addr** para um conjunto que contenha endereços ou faixas IPv6, como **2001:db8:1::1** ou **2001:db8:1::1/64**.
- **ether_addr** para um conjunto que contém uma lista de endereços de controle de acesso à mídia (MAC), como **52:54:00:6b:66:42**.

- **inet_proto** para um conjunto que contém uma lista de tipos de protocolos de Internet, como **tcp**.
- **inet_service** para um conjunto que contém uma lista de serviços de Internet, tais como **ssh**.
- **mark** para um conjunto que contém uma lista de marcas de pacotes. As marcas de pacotes podem ser qualquer valor inteiro positivo de 32 bits (**0** a **2147483647**].

Pré-requisitos

- A cadeia **example_chain** e a tabela **example_table** existem.

Procedimento

1. Criar um conjunto vazio. Os exemplos a seguir criam um conjunto para endereços IPv4:
   - Para criar um conjunto que possa armazenar múltiplos endereços IPv4 individuais:

     ```
     # nft add set inet example_table example_set { type ipv4_addr }; }
     ```

   - Para criar um conjunto que possa armazenar faixas de endereços IPv4:

     ```
     # nft add set inet example_table example_set { type ipv4_addr \; flags interval \; }
     ```

     IMPORTANTE

     Para evitar que a casca interprete os ponto-e-vírgula como o fim do comando, você deve escapar dos pontos-e-vírgula com uma barra invertida.

2. Opcionalmente, criar regras que utilizem o conjunto. Por exemplo, o seguinte comando adiciona uma regra ao **example_chain** no site **example_table** que irá descartar todos os pacotes de endereços IPv4 em **example_set**.

   ```
   # nft add rule inet example_table example_chain ip saddr @example_set drop
   ```

   Como **example_set** ainda está vazio, a regra atualmente não tem efeito.

3. Adicionar endereços IPv4 a **example_set**:
   - Se você criar um conjunto que armazene endereços IPv4 individuais, entre:

     ```
     # nft adicionar elemento inet example_table example_set { 192.0.2.1, 192.0.2.2 }
     ```

   - Se você criar um conjunto que armazene faixas IPv4, entre:

     ```
     # nft adicionar elemento inet example_table example_set { 192.0.2.0-192.0.2.255 }
     ```

     Quando você especifica uma faixa de endereços IP, você pode alternativamente usar a notação Classless Inter-Domain Routing (CIDR), como por exemplo **192.0.2.0/24** no exemplo acima.

6.5.3. Informações relacionadas

- Para mais detalhes sobre os conjuntos, consulte a seção **Sets** na página de manual **nft(8)**.

6.6. USANDO MAPAS DE VEREDICTOS EM COMANDOS NFTABLES

Os mapas verídicos, que também são conhecidos como dicionários, permitem que **nft** execute uma ação baseada em informações de pacotes, mapeando critérios de correspondência a uma ação.

6.6.1. Usando mapas literais em nftables

Um mapa literal é um **{ *match_criteria* : *action* }** declaração de que você usa diretamente em uma regra. A declaração pode conter vários mapeamentos separados por vírgula.

A desvantagem de um mapa literal é que se você quiser mudar o mapa, você deve substituir a regra. Para uma solução dinâmica, use mapas de veredictos nomeados, como descrito em Seção 6.6.2, "Usando mapas de veredictos mutáveis em nftables".

O exemplo descreve como usar um mapa literal para encaminhar tanto os pacotes TCP e UDP do protocolo IPv4 e IPv6 para diferentes cadeias a fim de contar separadamente os pacotes TCP e UDP que chegam.

Procedimento

1. Crie o **example_table**:

```
# nft adicionar tabela inet exemplo_tabela
```

2. Criar a cadeia **tcp_packets** em **example_table**:

```
# nft add chain inet exemplo_tabela tcp_packets
```

3. Adicione uma regra a **tcp_packets** que conta o tráfego nesta cadeia:

```
# nft add rule inet example_table tcp_packets counter
```

4. Crie a cadeia **udp_packets** em **example_table**

```
# nft add chain inet exemplo_tabela udp_packets
```

5. Adicione uma regra a **udp_packets** que conta o tráfego nesta cadeia:

```
# nft add rule inet example_table udp_packets counter
```

6. Criar uma cadeia para o tráfego de entrada. Por exemplo, para criar uma cadeia chamada **incoming_traffic** em **example_table** que filtra o tráfego de entrada:

```
# nft add chain inet example_table incoming_traffic { type filter hook input priority 0 { type filter hook input priority 0}; }
```

7. Adicione uma regra com um mapa literal a **incoming_traffic**:

```
# nft add rule inet example_table incoming_traffic ip protocol vmap { tcp : jump tcp_packets, udp : jump udp_packets }
```

O mapa literal distingue os pacotes e os envia para as diferentes cadeias de contadores com base em seu protocolo.

8. Para listar os balcões de trânsito, exibir **example_table**:

```
# nft list table inet example_table
table inet example_table {
  chain tcp_packets {
    counter packets 36379 bytes 2103816
  }

  chain udp_packets {
    counter packets 10 bytes 1559
  }

  chain incoming_traffic {
    type filter hook input priority filter; policy accept;
    ip protocol vmap { tcp : jump tcp_packets, udp : jump udp_packets }
  }
}
```

Os balcões da cadeia **tcp_packets** e **udp_packets** exibem tanto o número de pacotes recebidos quanto o número de bytes.

6.6.2. Usando mapas de veredictos mutáveis em nftables

A estrutura **nftables** suporta mapas de veredictos mutáveis. Você pode usar estes mapas em várias regras dentro de uma tabela. Outro benefício sobre os mapas literais é que você pode atualizar um mapa mutável sem substituir as regras que o utilizam.

Quando você cria um mapa de veredicto mutável, você deve especificar o tipo de elementos

- **ipv4_addr** para um mapa cuja parte correspondente contém um endereço IPv4, tal como **192.0.2.1**.
- **ipv6_addr** para um mapa cuja parte correspondente contém um endereço IPv6, tal como **2001:db8:1::1**.
- **ether_addr** para um mapa cuja parte correspondente contém um endereço de controle de acesso à mídia (MAC), tal como **52:54:00:6b:66:42**.
- **inet_proto** para um mapa cuja parte correspondente contém um tipo de protocolo Internet, tal como **tcp**.
- **inet_service** para um mapa cuja parte correspondente contém um número de porta do nome dos serviços da Internet, como **ssh** ou **22**.
- **mark** para um mapa cuja parte correspondente contém uma marca de pacote. Uma marca de pacote pode ser qualquer valor inteiro positivo de 32 bits (**0** a **2147483647**.
- **counter** para um mapa cuja parte correspondente contém um contravalor. O valor do contador pode ser qualquer valor inteiro positivo de 64 bits.
- **quota** para um mapa cuja parte correspondente contém um valor de cota. O valor da cota pode ser qualquer valor inteiro positivo de 64 bits.

O exemplo descreve como permitir ou largar pacotes de entrada com base em seu endereço IP de origem. Usando um mapa de veredicto mutável, é necessária apenas uma única regra para configurar este cenário enquanto os endereços IP e ações são armazenados dinamicamente no mapa. O

procedimento também descreve como adicionar e remover entradas do mapa.

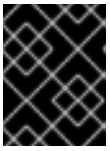
Procedimento

1. Criar uma mesa. Por exemplo, para criar uma tabela chamada **example_table** que processa pacotes IPv4:

```
# nft adicionar tabela ip exemplo_tabela
```

2. Criar uma corrente. Por exemplo, para criar uma cadeia chamada **example_chain** em **example_table**:

```
# nft add chain ip example_table example_chain { type filter hook input priority 0 {\i1}; {\i1}
```



IMPORTANTE

Para evitar que a casca interprete os ponto-e-vírgula como o fim do comando, você deve escapar dos pontos-e-vírgula com uma barra invertida.

3. Criar um mapa vazio. Por exemplo, para criar um mapa para endereços IPv4:

```
# nft add map ip example_table example_map { type ipv4_addr : veredicto }
```

4. Criar regras que utilizem o mapa. Por exemplo, o seguinte comando adiciona uma regra a **example_chain** em **example_table** que aplica ações a endereços IPv4 que são ambos definidos em **example_map**:

```
# nft add rule example_table example_chain ip saddr vmap @example_map
```

5. Adicionar endereços IPv4 e ações correspondentes a **example_map**:

```
# nft adicionar elemento ip example_table example_map { 192.0.2.1 : aceitar, 192.0.2.2 :
largar }
```

Este exemplo define os mapeamentos de endereços IPv4 para ações. Em combinação com a regra criada acima, o firewall aceita pacotes de **192.0.2.1** e deixa cair pacotes de **192.0.2.2**.

6. Opcionalmente, melhore o mapa adicionando outro endereço IP e declaração de ação:

```
# nft adicionar elemento ip example_table example_map { 192.0.2.3 : aceitar }
```

7. Opcionalmente, remova uma entrada do mapa:

```
# nft apagar elemento ip example_table example_map { 192.0.2.1 }
```

8. Opcionalmente, exibir o conjunto de regras:

```
# nft list ruleset
table ip example_table {
  map example_map {
    type ipv4_addr : verdict
    elements = { 192.0.2.2 : drop, 192.0.2.3 : accept }
```

```
  }

  chain example_chain {
    type filter hook input priority filter; policy accept;
    ip saddr vmap @example_map
  }
}
```

6.6.3. Informações relacionadas

- Para mais detalhes sobre os mapas de veredictos, consulte a seção **Maps** na página de manual **nft(8)**.

6.7. CONFIGURAÇÃO DO ENCAMINHAMENTO DE PORTAS USANDO NFTABLES

O redirecionamento de portas permite aos administradores encaminhar pacotes enviados a uma porta de destino específica para uma porta local ou remota diferente.

Por exemplo, se seu servidor web não tiver um endereço IP público, você pode definir uma regra de encaminhamento de porta em seu firewall que encaminha os pacotes recebidos na porta **80** e **443** no firewall para o servidor web. Com esta regra de firewall, os usuários na Internet podem acessar o servidor web usando o IP ou o nome do host do firewall.

6.7.1. Encaminhamento de pacotes de entrada para uma porta local diferente

Esta seção descreve um exemplo de como encaminhar pacotes IPv4 recebidos na porta **8022** para a porta **22** no sistema local.

Procedimento

1. Criar uma tabela com o nome **nat** com a família de endereços **ip**:

```
# nft adicionar tabela ip nat
```

2. Acrescente as cadeias **prerouting** e **postrouting** à tabela:

```
# nft -- adicionar prerouting de corrente ip nat { tipo nat hook prerouting priority -100 }; { tipo nat hook prerouting priority -100 }; { tipo nat hook prerouting priority -100 }
```

NOTA

Passe a opção **--** para o comando **nft** para evitar que a casca interprete o valor de prioridade negativa como uma opção do comando **nft**.

3. Adicionar uma regra à cadeia **prerouting** que redireciona os pacotes recebidos na porta **8022** para a porta local **22**:

```
# nft adicionar regra ip nat prerouting tcp dport 8022 redirecionar para :22
```

6.7.2. Encaminhamento de pacotes de entrada em uma porta local específica para um host diferente

Você pode usar uma regra de tradução de endereço de rede de destino (DNAT) para encaminhar pacotes de entrada em uma porta local para um host remoto. Isto permite aos usuários na Internet acessar um serviço que roda em um host com um endereço IP privado.

O procedimento descreve como encaminhar os pacotes IPv4 recebidos na porta local **443** para o mesmo número de porta no sistema remoto com o endereço IP **192.0.2.1**.

Pré-requisito

- Você está logado como o usuário **root** no sistema que deve encaminhar os pacotes.

Procedimento

1. Criar uma tabela com o nome **nat** com a família de endereços **ip**:

```
# nft adicionar tabela ip nat
```

2. Acrescente as cadeias **prerouting** e **postrouting** à tabela:

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

NOTA

Passe a opção **--** para o comando **nft** para evitar que a casca interprete o valor de prioridade negativa como uma opção do comando **nft**.

3. Adicione uma regra à cadeia **prerouting** que redireciona os pacotes recebidos na porta **443** para a mesma porta em **192.0.2.1**:

```
# nft adicionar regra ip nat prerouting tcp dport 443 dnat a 192.0.2.1
```

4. Adicione uma regra à cadeia **postrouting** para disfarçar o tráfego de saída:

```
# nft adicionar regra ip daddr 192.0.2.1 mascarada
```

5. Habilitar o envio de pacotes:

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

6.8. UTILIZAÇÃO DE NFTABLES PARA LIMITAR A QUANTIDADE DE CONEXÕES

Você pode usar **nftables** para limitar o número de conexões ou para bloquear endereços IP que tentam estabelecer uma determinada quantidade de conexões para evitar que elas utilizem muitos recursos do sistema.

6.8.1. Limitando o número de conexões usando nftables

O parâmetro **ct count** do utilitário **nft** permite aos administradores limitar o número de conexões. O procedimento descreve um exemplo básico de como limitar as conexões de entrada.

Pré-requisitos

- A base **example_chain** em **example_table** existe.

Procedimento

1. Adicione uma regra que permite apenas duas conexões simultâneas à porta SSH (22) a partir de um endereço IPv4 e rejeita todas as outras conexões a partir do mesmo IP:

```
# nft add rule ip example_table example_chain tcp dport ssh meter example_meter { ip saddr ct count over 2 } counter reject
```

2. Opcionalmente, exibir o medidor criado na etapa anterior:

```
# nft list meter ip example_table example_meter
table ip example_table {
  meter example_meter {
    type ipv4_addr
    size 65535
    elements = { 192.0.2.1 : ct count over 2 , 192.0.2.2 : ct count over 2  }
  }
}
```

A entrada **elements** exibe endereços que atualmente correspondem à regra. Neste exemplo, **elements** lista os endereços IP que têm conexões ativas com a porta SSH. Observe que a saída não exibe o número de conexões ativas ou se as conexões foram rejeitadas.

6.8.2. Bloqueio de endereços IP que tentam mais de dez novas conexões TCP de entrada em um minuto

A estrutura **nftables** permite que os administradores atualizem dinamicamente os conjuntos. Esta seção explica como usar esta funcionalidade para bloquear temporariamente hosts que estão estabelecendo mais de dez conexões TCP IPv4 dentro de um minuto. Após cinco minutos, **nftables** remove automaticamente o endereço IP da lista de negação.

Procedimento

1. Criar a tabela **filter** com a família de endereços **ip**:

```
# nft adicionar tabela ip filter
```

2. Acrescente a cadeia **input** à tabela **filter**:

```
# nft add chain ip filter input { type filter hook input priority 0 { type hook input priority 0}; { type filter hook input priority 0}; { type filter hook input priority 0
```

3. Adicione um conjunto chamado **denylist** à tabela **filter**:

```
# nft add set ip filter denylist { type ipv4_addr }; flags dynamic, timeout; timeout 5m; timeout
```

Este comando cria um conjunto dinâmico para endereços IPv4. O parâmetro **timeout 5m** define que **nftables** remove automaticamente as entradas após 5 minutos do conjunto.

4. Adicionar uma regra que automaticamente adiciona o endereço IP de origem dos hosts que tentam estabelecer mais de dez novas conexões TCP dentro de um minuto ao conjunto **denylist**:

```
# nft add rule ip filter input ip protocol tcp ct state new, unracked limit rate over 10/minute add @denylist { ip saddr }
```

5. Acrescente uma regra que abandone todas as conexões de endereços IP no conjunto **denylist**:

```
# nft add rule ip filter input ip saddr @denylist drop
```

Recursos adicionais

- Seção 6.5.2, "Usando conjuntos nomeados em nftables"

6.9. REGRAS DE DEPURAÇÃO DE NFTABLES

A estrutura **nftables** oferece diferentes opções para os administradores depurarem as regras e se os pacotes corresponderem a elas. Esta seção descreve estas opções.

6.9.1. Criando uma regra com um contador

Para identificar se uma regra é igualada, você pode usar um contador. Esta seção descreve como criar uma nova regra com um contador.

Para um procedimento que acrescenta um contrário a uma regra existente, ver Seção 6.9.2, "Adicionando um contador a uma regra existente".

Pré-requisitos

- A cadeia à qual se deseja acrescentar a regra existe.

Procedimento

1. Adicione uma nova regra com o parâmetro **counter** à cadeia. O exemplo a seguir adiciona uma regra com um contador que permite o tráfego TCP na porta 22 e conta os pacotes e o tráfego que correspondem a esta regra:

```
# nft adicionar regra inet example_table example_chain tcp dport 22 counter accept
```

2. Para exibir os valores do contador:

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

```
    tcp dport ssh counter packets 6872 bytes 105448565 accept
  }
}
```

6.9.2. Adicionando um contador a uma regra existente

Para identificar se uma regra é igualada, você pode usar um contador. Esta seção descreve como adicionar um contador a uma regra existente.

Para um procedimento para adicionar uma nova regra com um contador, ver Seção 6.9.1, “Criando uma regra com um contador”.

Pré-requisitos

- A regra à qual se deseja acrescentar o contador existe.

Procedimento

1. Mostrar as regras da corrente incluindo seus cabos:

```
# nft --handle list chain inet example_table example_chain
table inet example_table {
  chain example_chain { # handle 1
    type filter hook input priority filter; policy accept;
    tcp dport ssh accept # handle 4
  }
}
```

2. Adicione o contador substituindo a regra, mas com o parâmetro **counter**. O exemplo a seguir substitui a regra exibida na etapa anterior e adiciona um contador:

```
# nft replace rule inet example_table example_chain handle 4 tcp dport 22 counter accept
```

3. Para exibir os valores do contador:

```
# nft list ruleset
table inet example_table {
  chain example_chain {
    type filter hook input priority filter; policy accept;
    tcp dport ssh counter packets 6872 bytes 105448565 accept
  }
}
```

6.9.3. Pacotes de monitoramento que correspondem a uma regra existente

O recurso de rastreamento em **nftables** em combinação com o comando **nft monitor** permite que os administradores exibam pacotes que correspondem a uma regra. O procedimento descreve como permitir o rastreamento de uma regra, bem como o monitoramento de pacotes que correspondam a esta regra.

Pré-requisitos

- A regra à qual se deseja acrescentar o contador existe.

Procedimento

1. Mostrar as regras da corrente incluindo seus cabos:

```
# nft --handle list chain inet example_table example_chain
table inet example_table {
  chain example_chain { # handle 1
    type filter hook input priority filter; policy accept;
    tcp dport ssh accept # handle 4
  }
}
```

2. Adicione o recurso de rastreamento substituindo a regra, mas com os parâmetros **meta nftrace set 1**. O exemplo a seguir substitui a regra exibida na etapa anterior e permite o rastreamento:

```
# nft replace rule inet example_table example_chain handle 4 tcp dport 22 meta nftrace set 1 accept
```

3. Use o comando **nft monitor** para exibir o rastreamento. O seguinte exemplo filtra a saída do comando para exibir somente as entradas que contenham **inet example_table example_chain**:

```
# nft monitor | grep "inet example_table example_chain"
trace id 3c5eb15e inet example_table example_chain packet: iif "enp1s0" ether saddr 52:54:00:17:ff:e4 ether daddr 52:54:00:72:2f:6e ip saddr 192.0.2.1 ip daddr 192.0.2.2 ip dscp cs0 ip ecn not-ect ip ttl 64 ip id 49710 ip protocol tcp ip length 60 tcp sport 56728 tcp dport ssh tcp flags == syn tcp window 64240
trace id 3c5eb15e inet example_table example_chain rule tcp dport ssh nftrace set 1 accept (verdict accept)
...
```

ATENÇÃO

Dependendo do número de regras com rastreamento habilitado e da quantidade de tráfego correspondente, o comando **nft monitor** pode exibir uma grande quantidade de resultados. Use **grep** ou outras utilidades para filtrar a saída.

6.10. APOIO E RESTAURAÇÃO DOS CONJUNTOS DE REGRAS NFTABLES

Esta seção descreve como fazer backup das regras **nftables** em um arquivo, assim como restaurar as regras de um arquivo.

Os administradores podem usar um arquivo com as regras para, por exemplo, transferir as regras para um servidor diferente.

6.10.1. Cópia de segurança dos conjuntos de regras nftables para um arquivo

Esta seção descreve como fazer backup do conjunto de regras **nftables** para um arquivo.

Procedimento

- Para fazer backup das regras **nftables**:
  - No formato **nft list ruleset**:

```
# nft list ruleset > file.nft
```

  - No formato JSON:

```
# nft -j list ruleset > file.json
```

6.10.2. Restauração de conjuntos de regras nftables a partir de um arquivo

Esta seção descreve como restaurar os conjuntos de regras **nftables**.

Procedimento

- Para restaurar as regras do **nftables**:
  - Se o arquivo a ser restaurado estiver no formato **nft list ruleset** ou contiver comandos **nft**:

```
# nft -f file.nft
```

  - Se o arquivo a ser restaurado estiver no formato JSON:

```
# nft -j -f file.json
```

6.11. INFORMAÇÕES RELACIONADAS

- O post Usando nftables no blog Red Hat Enterprise Linux 8 fornece uma visão geral sobre o uso dos recursos do **nftables**.
- O que vem depois do iptables? Seu sucessor, é claro: o artigo nftables explica porque **nftables** substitui **iptables**.
- O Firewalld: The Future is nftables article provides additional information on **nftables** as a default back end for **firewalld**.