

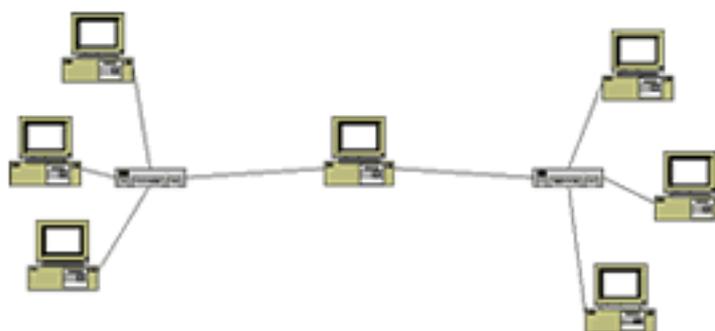
1. ROTEAMENTO DINÂMICO

Considerações iniciais

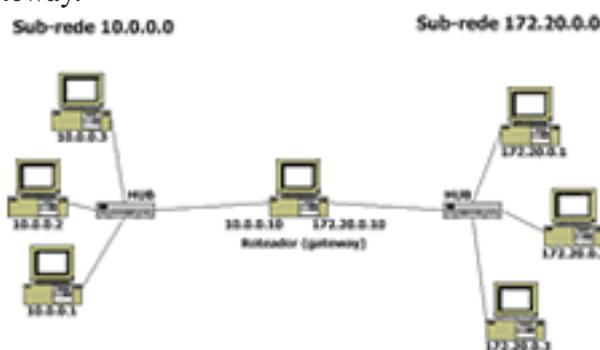
O roteamento dinâmico é estabelecido quando colocamos uma máquina entre duas ou mais sub-redes diferentes e há a livre passagem de pacotes entre elas, quando necessário. É importante ressaltar que o roteamento só irá funcionar quando for feito entre SUB-REDES DIFERENTES. Não se pode colocar um roteador entre duas sub-redes iguais.

O roteamento também é útil para diminuir o tráfego na rede como um todo, pois só deixa o pacote mudar de sub-rede se isso for realmente necessário.

Veja um esquema de roteamento:



Para fazer o roteamento, o micro roteador deve possuir uma placa de rede em cada sub-rede. Também deveremos informar em cada máquina quem será o micro responsável pelo roteamento. O nome técnico desse micro é gateway.



No caso do esquema anterior, temos as seguintes situações:

* Gateway para 10.0.0.1, 10.0.0.2 e 10.0.0.3: 10.0.0.10

* Gateway para 172.20.0.1, 172.20.0.2 e 172.20.0.3: 172.20.0.10

É importante que, nos dois lados, todos os micros saibam quem é o seu gateway pois, do contrário, os pacotes poderão se perder na ida ou na volta. Exemplo:

A máquina 10.0.0.1 sabe que 10.0.0.10 é o seu gateway. A máquina 172.20.0.1 não sabe quem é o seu gateway. Um ping de 10.0.0.1 para 172.20.0.1 sairá de 10.0.0.1, passará por 10.0.0.10, seguirá por 172.20.0.10 e chegará em 172.20.0.1. Como o 172.20.0.1 não sabe quem é o seu gateway para a rede 10.0.0.0, não conseguirá responder. O ping vai morrer por timeout. Houve o icmp echo request, mas será impossível gerar o icmp echo reply.

Fazendo o roteamento dinâmico

Para estabelecermos o roteamento dinâmico entre duas sub-redes, basta:

- * inserir um micro com duas placas de rede entre as duas sub-redes, configurando cada placa de acordo com cada sub-rede;
- * definir, em cada máquina, de cada sub-rede, quem é o seu gateway;
- * ativar o roteamento dinâmico via kernel.

Definindo o gateway

Para definirmos o gateway em cada cliente, devemos:

--> **No Linux**

Editar o arquivo `/etc/sysconfig/network` e inserir a linha:

```
GATEWAY=ip_do_gateway
```

Em seguida, devemos reiniciar a rede:

```
#/etc/rc.d/init.d/network restart
```

--> **No Windows**

Nas Propriedades do protocolo TCP/IP, há uma seção gateway. Basta inserir o IP do gateway nessa seção. Exemplo:



Ativando o roteamento via kernel

O roteamento via kernel será ativado com o comando:

```
#echo 1 > /proc/sys/net/ipv4/ip_forward
```

Esse roteamento será perdido se a rede (e, em consequência, a máquina) for reinicializada (`#/etc/rc.d/init.d/network restart`).

Poderíamos inserir a regra no fim do arquivo `/etc/rc.d/rc.local`, para que a mesma seja ativada a cada reinicialização do sistema. No entanto, um reinício da rede mataria o roteamento novamente.

Uma forma de deixar a regra de roteamento permanentemente ativada, resistindo a qualquer tipo de reinicialização, seria a alteração do arquivo `/etc/sysctl.conf`:

```
net.ipv4.ip_forward = 1
```

Teste do roteamento

O teste do roteamento pode ser feito por intermédio do comando `ping`. Basta “pingar” uma máquina que esteja após o roteador.

Caso não haja resposta, faça um teste progressivo para tentar deduzir o problema:

--> pingue a placa do roteador que esteja dentro da sua sub-rede;

--> pingue a placa do roteador que esteja na outra sub-rede;

--> pingue uma máquina da outra sub-rede.

Roteamento múltiplo

É possível fazer um roteamento múltiplo, entre mais de duas sub-redes. Basta acrescentar mais placas de rede no roteador e executar os mesmos procedimentos aqui descritos.

Cabe ressaltar que nem todo micro, dependendo do seu hardware, aceita mais de duas placas de rede.

2. GENERALIDADES

Considerações iniciais

Existem vários tipos de firewall. Entende-se por firewall como sendo qualquer máquina capaz de tomar decisões em relação ao tráfego de rede. Podemos citar como tipos de firewall, as máquinas que executam os seguintes serviços:

--> roteamento controlado por regras de análise de cabeçalho IP (filtro de pacotes ou firewall de passagem);

--> roteamento mascarado controlado por regras de análise de cabeçalho IP (filtro de pacotes mascarado ou firewall de mascaramento);

--> roteamento controlado por regras de análise de conteúdo de pacotes (filtro de análise de pacotes ou firewall de conteúdo);

--> roteamento mascarado controlado por regras de análise de URL (proxy).

Falarei sobre o filtro de pacotes existente no Linux. Ele verifica apenas o cabeçalho de cada pacote, definindo o que ocorrerá com tais pacotes. Basicamente, só entende endereço IP, máscara de sub-rede, portas e tipos de protocolos. Não analisa o conteúdo do pacote.

Todas as expressões firewall, quando utilizadas daqui por diante, referir-se-ão ao filtro de pacotes do Linux.

A filtragem de pacotes é uma atividade interna do kernel.

Os filtros Linux

O firewall, na maioria das vezes, atua como um roteamento dinâmico controlado. É uma atividade interna, uma propriedade, do kernel.

São os seguintes, os filtros existentes:

--> kernel 2.0.x: ipfwadm

--> kernel 2.2.x: ipchains

--> kernel 2.4.x: iptables

Obs: um kernel é estável quando o algarismo existente entre os dois pontos é par. Exemplo: x.2.x é estável. Já o x.3.x não é estável.

O Kernel 2.4, por questões de compatibilidade, mantém os filtros ipfwadm e ipchains. No entanto, eles não funcionam completamente com esse kernel. Além disso, se o ipchains estiver ativo, o iptables não irá funcionar. Assim, no Red Hat, torna-se necessário entrar no #setup e:

--> habilitar o iptables

--> desabilitar o ipchains

Se a distribuição utilizada não possuir o comando #setup, utilize o comando #rmmod ipchains. Cabe ressaltar que o iptables terá os seus módulos básicos carregados quando for utilizado pela primeira vez.

Como funciona um firewall ?

O FILTRO DE PACOTES do Linux funciona mediante regras estabelecidas. Todos os pacotes entram no kernel para serem analisados. As CHAINS (correntes) são as situações possíveis dentro do kernel. Quando um pacote entra no kernel, este verifica o destino do pacote e decide qual chain irá tratar do pacote. Isso se chama roteamento interno. Os tipos de chains irão depender da tabela que estaremos utilizando no momento. Existem 3 tabelas possíveis:

--> filter: é a tabela default. Quando não especificarmos a tabela, a filter será utilizada. Refere-se às atividades normais de tráfego de dados, sem a ocorrência de NAT. Admite as chains INPUT, OUTPUT e FORWARD.

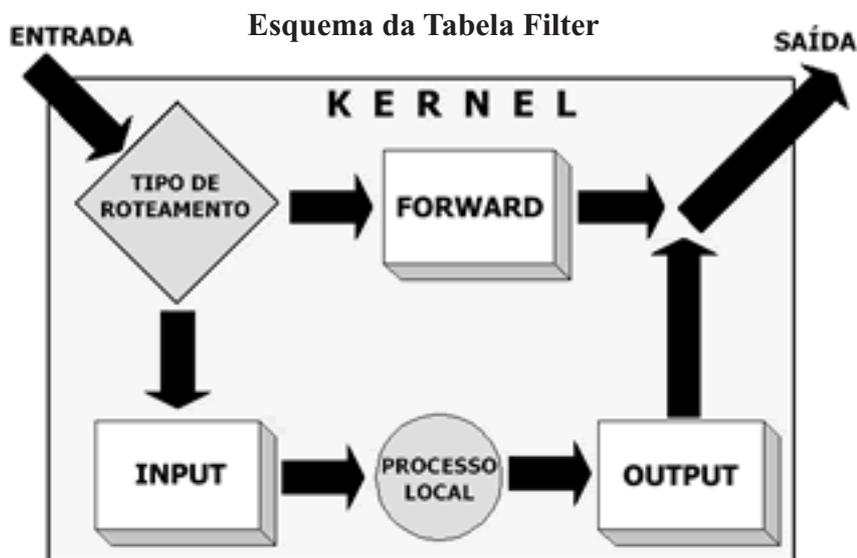
--> nat: utilizada quando há NAT. Exemplo: passagem de dados de uma rede privada para a Internet. Admite as chains PREROUTING, OUTPUT e POSTROUTING.

--> mangle: há referências de que a mesma é utilizada para alterações especiais em pacotes. Raramente utilizada.

3. TABELAS

Tabela Filter

Vejam os funcionamento da tabela filter (default) e as suas respectivas chains:



São três, as possíveis chains:

--> INPUT: utilizada quando o destino final é a própria máquina firewall;

--> OUTPUT: qualquer pacote gerado na máquina firewall e que deva sair para a rede será tratado pela chain OUTPUT;

--> FORWARD: qualquer pacote que atravessa o firewall, oriundo de uma máquina e direcionado a outra, será tratado pela chain FORWARD.

Regras de firewall

As regras (rules) de firewall, geralmente, são compostas assim:

```
#iptables [-t tabela] [opção] [chain] [dados] -j [ação]
```

Exemplo:

```
#iptables -A FORWARD -d 192.168.1.1 -j DROP
```

A linha acima determina que todos os pacotes destinados à máquina 192.168.1.1 devem ser descartados. No caso:

tabela: filter (é a default)

opção: -A

chain: FORWARD

dados: -d 192.168.1.1

ação: DROP

Existem outras possibilidades que fogem à sintaxe mostrada anteriormente. É o caso do comando #iptables -L, que mostra as regras em vigor.

Análise de regras com a tabela filter

Opções

As principais opções são:

-P --> Policy (política). Altera a política da chain. A política inicial de cada chain é ACCEPT. Isso faz com que o firewall, inicialmente, aceite qualquer INPUT, OUTPUT ou FORWARD. A política pode ser alterada para DROP, que irá negar o serviço da chain, até que uma opção **-A** entre em vigor. O **-P** não aceita REJECT ou LOG. Exemplos:

```
#iptables -P FORWARD DROP
```

```
#iptables -P INPUT ACCEPT
```

-A --> Append (anexar). Acresce uma nova regra à chain. Tem prioridade sobre o **-P**. Geralmente, como buscamos segurança máxima, colocamos todas as chains em política DROP, com o **-P** e, depois, abrimos o que é necessário com o **-A**. Exemplos:

```
#iptables -A OUTPUT -d 172.20.5.10 -j ACCEPT
```

```
#iptables -A FORWARD -s 10.0.0.1 -j DROP
```

```
#iptables -A FORWARD -d www.chat.com.br -j DROP
```

-D --> Delete (apagar). Apaga uma regra. A regra deve ser escrita novamente, trocando-se a opção para **-D**. Exemplos:

Para apagar as regras anteriores, usa-se:

```
#iptables -D OUTPUT -d 172.20.5.10 -j ACCEPT
```

```
#iptables -D FORWARD -s 10.0.0.1 -j DROP
```

```
#iptables -D FORWARD -d www.chat.com.br -j DROP
```

Também é possível apagar a regra pelo seu número de ordem. Pode-se utilizar o **-L** para verificar o número de ordem. Verificado esse número, basta citar a chain e o número de ordem. Exemplo:

```
#iptables -D FORWARD 4
```

Isso deleta a regra número 4 de forward.

-L --> List (listar). Lista as regras existentes. Exemplos:

```
#iptables -L
```

```
#iptables -L FORWARD
```

-F --> Flush (esvaziar). Remove todas as regras existentes. No entanto, não altera a política (**-P**). Exemplos:

```
#iptables -F
```

```
#iptables -F FORWARD
```

Chains

As chains já são conhecidas:

INPUT --> Refere-se a todos os pacotes destinados à máquina firewall.

OUTPUT --> Refere-se a todos os pacotes gerados na máquina firewall.

FORWARD --> Refere-se a todos os pacotes oriundos de uma máquina e destinados a outra. São pacotes que atravessam a máquina firewall, mas não são destinados a ela.

Dados

Os elementos mais comuns para se gerar dados são os seguintes:

-s --> Source (origem). Estabelece a origem do pacote. Geralmente é uma combinação do endereço IP com a máscara de sub-rede, separados por uma barra. Exemplo:

```
-s 172.20.0.0/255.255.0.0
```

No caso, vimos a sub-rede 172.20.0.0. Para hosts, a máscara sempre será 255.255.255.255. Exemplo:

```
-s 172.20.5.10/255.255.255.255
```

Agora vimos o host 172.20.5.10. Ainda no caso de hosts, a máscara pode ser omitida. Caso isso ocorra, o iptables considera a máscara como 255.255.255.255. Exemplo:

```
-s 172.20.5.10
```

Isso corresponde ao host 172.20.5.10. Há um recurso para simplificar a utilização da máscara de sub-rede. Basta utilizar a quantidade de bits 1 existentes na máscara. Assim, a máscara 255.255.0.0 vira 16. A utilização fica assim:

```
-s 172.20.0.0/16
```

Outra possibilidade é a designação de hosts pelo nome. Exemplo:

```
-s www.chat.com.br
```

Para especificar qualquer origem, utilize a rota default, ou seja, 0.0.0.0/0.0.0.0, também admitindo 0/0.

-d --> Destination (destino). Estabelece o destino do pacote. Funciona exatamente como o -s, incluindo a sintaxe.

-p --> Protocol (protocolo). Especifica o protocolo a ser filtrado. O protocolo IP pode ser especificado pelo seu número (vide /etc/protocols) ou pelo nome. Os protocolos mais utilizados são udp, tcp e icmp. Exemplo:

```
-p icmp
```

-i --> In-Interface (interface de entrada). Especifica a interface de entrada. As interfaces existentes podem ser vistas com o comando #ifconfig. O -i não pode ser utilizado com a chain OUTPUT. Exemplo:

```
-i ppp0
```

O sinal + pode ser utilizado para simbolizar várias interfaces. Exemplo:

-i eth+

eth+ refere-se à eth0, eth1, eth2 etc.

-o --> Out-Interface (interface de saída). Especifica a interface de saída. Similar a -i, inclusive nas flexibilidades. O -o não pode ser utilizado com a chain INPUT.

! --> Exclusão. Utilizado com -s, -d, -p, -i, -o e outros, para excluir o argumento. Exemplo:

-s ! 10.0.0.1

Isso refere-se a qualquer endereço de entrada, exceto o 10.0.0.1.

-p ! tcp

Todos os protocolos, exceto o TCP.

--sport --> Source Port. Porta de origem. Só funciona com as opções -p udp e -p tcp. Exemplo:

-p tcp --sport 80

Refere-se à porta 80 sobre protocolo TCP.

--dport --> Destination Port. Porta de destino. Só funciona com as opções -p udp e -p tcp. Similar a --sport.

Ações

As principais ações são:

ACCEPT --> Aceitar. Permite a passagem do pacote.

DROP --> Abandonar. Não permite a passagem do pacote, descartando-o. Não avisa a origem sobre o ocorrido.

REJECT --> Igual ao DROP, mas avisa a origem sobre o ocorrido (envia pacote icmp unreachable).

LOG --> Cria um log referente à regra, em /var/log/messages. Usar antes de outras ações.

Exemplos comentados de regras de firewall (tabela filter)

#iptables -L

Lista todas as regras existentes.

#iptables -F

Apaga todas as regras sem alterar a política.

#iptables -P FORWARD DROP

Estabelece uma política de proibição inicial de passagem de pacotes entre sub-redes.

#iptables -A FORWARD -j DROP

Todos os pacotes oriundos de qualquer sub-rede e destinados a qualquer sub-rede deverão ser descartados.

#iptables -A FORWARD -j ACCEPT

Todos os pacotes oriundos de qualquer sub-rede e destinados a qualquer sub-rede deverão ser aceitos.

#iptables -A FORWARD -s 10.0.0.0/8 -d www.chat.com.br -j DROP

Os pacotes oriundos da sub-rede 10.0.0.0 (máscara 255.0.0.0) e destinados ao host www.chat.com.br deverão ser descartados.

#iptables -A FORWARD -s 10.0.0.0/8 -d www.chat.com.br -j REJECT

Os pacotes oriundos da sub-rede 10.0.0.0 (máscara 255.0.0.0) e destinados ao host www.chat.com.br deverão ser descartados. Deverá ser enviado um ICMP avisando à origem.

#iptables -A FORWARD -d www.chat.com.br -j DROP

Os pacotes oriundos de qualquer lugar e destinados ao host www.chat.com.br deverão ser descartados.

#iptables -A FORWARD -d 10.0.0.0/8 -s www.chat.com.br -j DROP

Os pacotes destinados à sub-rede 10.0.0.0 (máscara 255.0.0.0) e oriundos do host www.chat.com.br deverão ser descartados.

#iptables -A FORWARD -s www.chat.com.br -j DROP

Os pacotes oriundos do host www.chat.com.br e destinados a qualquer lugar deverão ser descartados.

#iptables -A FORWARD -s 200.221.20.0/24 -j DROP

Os pacotes oriundos da sub-rede 200.221.20.0 (máscara 255.255.255.0) e destinados a qualquer lugar deverão ser descartados.

#iptables -A FORWARD -s 10.0.0.5 -p icmp -j DROP

Os pacotes icmp oriundos do host 10.0.0.5 e destinados a qualquer lugar deverão ser descartados.

#iptables -A FORWARD -i eth0 -j ACCEPT

Os pacotes que entrarem pela interface eth0 serão aceitos.

#iptables -A FORWARD -i ! eth0 -j ACCEPT

Os pacotes que entrarem por qualquer interface, exceto a eth0, serão aceitos.

#iptables -A FORWARD -s 10.0.0.5 -p tcp --sport 80 -j LOG

O tráfego de pacotes TCP oriundos da porta 80 do host 10.0.0.5 e destinados a qualquer lugar deverá ser gravado em log. No caso, /var/log/messages.

```
#iptables -A FORWARD -p tcp --dport 25 -j ACCEPT
```

Os pacotes TCP destinados à porta 25 de qualquer host deverão ser aceitos.

Observações importantes

Impasses e ordem de processamento

Se houver impasse entre regras, sempre valerá a primeira. Assim, entre as regras:

```
#iptables -A FORWARD -p icmp -j DROP
```

```
#iptables -A FORWARD -p icmp -j ACCEPT
```

Valerá:

```
#iptables -A FORWARD -p icmp -j DROP
```

Já entre as regras:

```
#iptables -A FORWARD -p icmp -j ACCEPT
```

```
#iptables -A FORWARD -p icmp -j DROP
```

Valerá:

```
#iptables -A FORWARD -p icmp -j ACCEPT
```

Isso ocorre porque as regras são processadas na ordem em que aparecem. Depois do processamento da regra, pode haver continuidade de processamento ou não. Isso irá depender da ação:

ACCEPT --> Pára de processar regras para o pacote atual;

DROP --> Pára de processar regras para o pacote atual;

REJECT --> Pára de processar regras para o pacote atual;

LOG --> Continua a processar regras para o pacote atual;

O retorno

Ao se fazer determinadas regras, devemos prever o retorno. Assim, digamos que exista a seguinte situação:

```
#iptables -P FORWARD DROP
```

```
#iptables -A FORWARD -s 10.0.0.0/8 -d 172.20.0.0/16 -j ACCEPT
```

Com as regras anteriores, fechamos todo o FORWARD e depois abrimos da sub-rede 10.0.0.0 para a sub-rede 172.20.0.0. No entanto, não tornamos possível a resposta da sub-rede 172.20.0.0 para a sub-rede 10.0.0.0. O correto, então, seria:

```
#iptables -P FORWARD DROP
```

```
#iptables -A FORWARD -s 10.0.0.0/8 -d 172.20.0.0/16 -j ACCEPT
```

```
#iptables -A FORWARD -d 10.0.0.0/8 -s 172.20.0.0/16 -j ACCEPT
```

Roteamento dinâmico

Caso haja o envolvimento de mais de uma sub-rede, será necessário que o roteamento dinâmico seja ativado para que o iptables funcione corretamente. O roteamento dinâmico, via kernel, pode ser ativado pelo comando:

```
#echo 1 > /proc/sys/net/ipv4/ip_forward
```

Cabe lembrar que a reinicialização do daemon de rede fará com que o roteamento dinâmico seja perdido. Uma forma de deixar a regra de roteamento permanentemente ativada, resistindo a qualquer tipo de reinicialização, seria a alteração do arquivo /etc/sysctl.conf:

```
net.ipv4.ip_forward = 1
```

O carregamento

Na primeira vez em que o iptables for utilizado, poderá surgir a mensagem:

```
ip_tables: (c)2000 Netfilter core team
```

Não se preocupe. Isso é normal. É o carregamento dos módulos necessários. Caso surjam mensagens de erro, certifique-se de que o ipchains não esteja carregado. Caso esteja, descarregue-o:

```
#rmmod ipchains
```

Salvando e recuperando tudo

As regras iptables poderão ser salvas com o comando:

```
#iptables-save > arquivo
```

A recuperação poderá ser feita pelo comando:

```
#iptables-restore < arquivo
```

Um típico exemplo de carregamento de regras de iptables, após a inicialização do sistema, seria:

```
#echo 1 > /proc/sys/net/ipv4/ip_forward
```

```
#iptables-restore < /etc/firewall
```

Isso pode ser inserido no fim do arquivo `/etc/rc.d/rc.local`.

Extensões

As extensões permitem filtragens especiais, principalmente contra ataques de hackers. Quando necessárias, devem ser as primeiras linhas do firewall. As mais importantes são:

Contra Ping

```
#iptables -A FORWARD -p icmp --icmp-type echo-request -j DROP
```

Contra Ping of Death

```
#iptables -A FORWARD -p icmp --icmp-type echo-request -m limit --limit 1/s -j ACCEPT
```

Contra ataques Syn-flood

```
#iptables -A FORWARD -p tcp -m limit --limit 1/s -j ACCEPT
```

Contra Port scanners avançados (nmap)

```
#iptables -A FORWARD -p tcp --tcp-flags SYN,ACK,FIN,RST -m limit --limit 1/s -j ACCEPT
```

Mais proteção

Existe, ainda, uma regra muito importante que não é extensão mas também pode ser utilizada como segurança. É a proteção contra pacotes danificados ou suspeitos.

```
#iptables -A FORWARD -m unclean -j DROP
```

Network Address Translator (tabela nat)

Existem vários recursos que utilizam NAT. Os mais conhecidos são:

--> Mascaramento (masquerading)

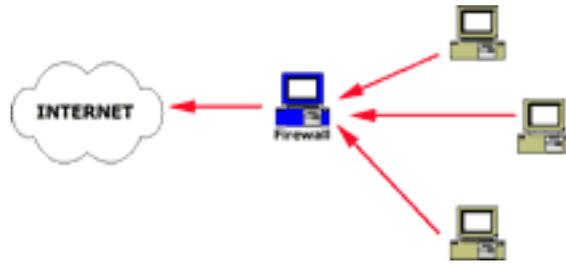
--> Redirecionamento de portas (port forwarding)--> Redirecionamento de servidores (forwarding)

--> Proxy transparente (transparent proxy)

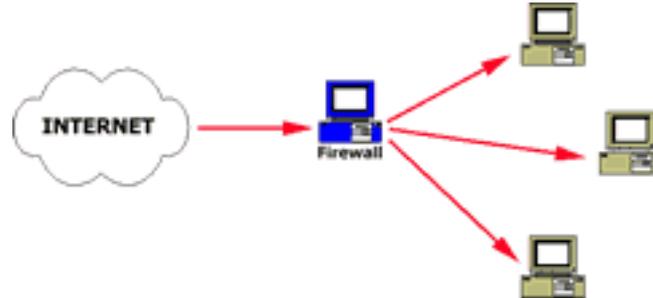
--> Balanceamento de carga (load balance)

Mascaramento

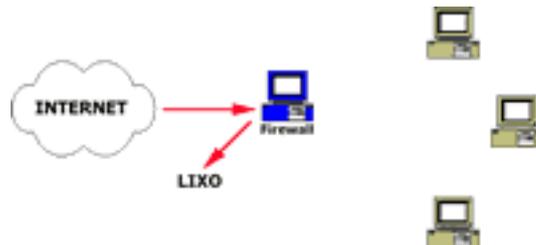
O mascaramento é uma forma de fazer NAT (Network Address Translation). Com isso, é possível fazer uma rede inteira navegar na Internet com segurança. A rede solicita os dados para a máquina que faz o mascaramento. Essa busca tais dados na Internet...



...e os entrega aos solicitantes:



No entanto, um host da Internet, por vontade própria, não consegue ultrapassar o firewall que faz mascaramento, em direção à rede:



O único endereço IP que irá circular na Internet será o do firewall.

O mascaramento também possui um esquema de funcionamento. Como haverá trocas de endereços, deveremos utilizar a tabela NAT para fazer isso.

Redirecionamento de portas

O redirecionamento de portas ocorre quando desejamos alterar a porta de destino de uma requisição.

Exemplo: tudo que for destinado à porta 23 de qualquer máquina, quando passar pela máquina firewall, será redirecionado para a porta 10000 de outro server.

Redirecionamento de servidores

Todos os pacotes destinados a um servidor ou porta do mesmo, serão redirecionados para outro servidor ou porta de outro servidor.

Proxy transparente

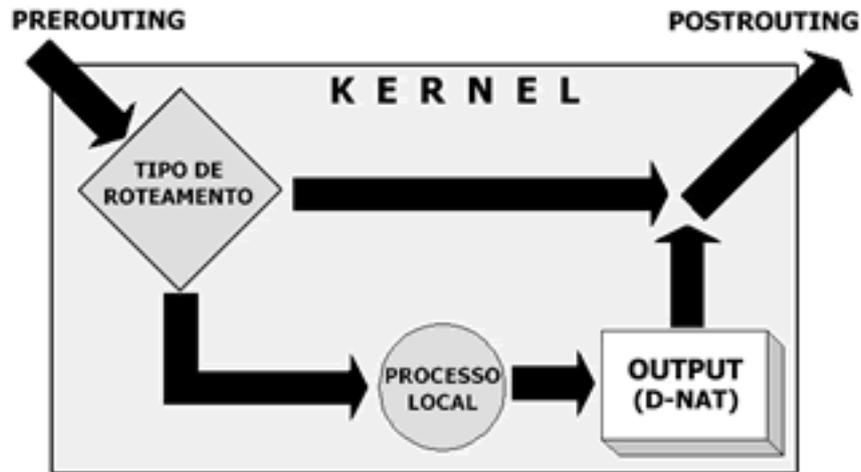
É a técnica que força o uso de um servidor proxy na rede.

Balanceamento de carga

O balanceamento de carga (load balance) é uma técnica utilizada para distribuir carga em clusters servidores. Entende-se por cluster, uma série de servidores grupados e sincronizados, a fim de conterem os mesmos dados. O load balance é o ato de distribuir os clientes aos servidores mais desocupados. Esse trabalho também pode ser feito por servidores DNS.

A tabela NAT

A tabela NAT funciona da seguinte forma:



O NAT é dividido em:

--> SNAT: aplica-se quando desejamos alterar o endereço de origem do pacote. Somente a chain POSTROUTING faz SNAT. O mascaramento é um exemplo de SNAT.

--> DNAT: aplica-se quando desejamos alterar o endereço de destino do pacote. As chains PREROUTING e OUTPUT fazem DNAT. O redirecionamento de porta, o redirecionamento de servidor, o load balance e o proxy transparente são exemplos de DNAT.

Regras de NAT

Para fazer o mascaramento, deveremos, antes, carregar o módulo de NAT:

```
#modprobe iptable_nat
```

As regras mais utilizadas, além da maioria dos recursos descritos para uso com a tabela filter, contêm o seguinte:

Chains

Existem as seguintes chains:

--> PREROUTING: utilizada para analisar pacotes que estão entrando no kernel para sofrerem NAT. O PREROUTING pode fazer ações de NAT com o endereço de destino do pacote. Isso é conhecido como DNAT (Destination NAT);

--> POSTROUTING: utilizada para analisar pacotes que estão saindo do kernel, após sofrerem NAT. O POSTROUTING pode fazer ações de NAT com o endereço de origem do pacote. Isso é conhecido como SNAT (Source NAT);

--> OUTPUT: utilizada para analisar pacotes que são gerados na própria máquina e que irão sofrer NAT. O OUTPUT pode fazer ações de NAT com o endereço de destino do pacote. Também é DNAT.

Opções

-A --> Append (anexar).

-D --> Deletar.

Dados

-t --> Table (tabela). Estabelece a tabela a ser utilizada. A tabela default, por omissão, é filter. Para o mascaramento ou NAT será nat. Exemplo:

```
#iptables -t nat -A ...
```

--to --> utilizado para definir IP e porta de destino, após um DNAT, ou de origem, após um SNAT. Deve ser utilizado após uma ação (-j ação). Assim:

```
-j DNAT --to 10.0.0.2
```

```
-j DNAT --to 10.0.0.2:80
```

```
-j SNAT --to 172.20.0.2
```

--dport --> assim como -d define um host de destino, --dport define uma porta de destino. Deve ser utilizado antes de uma ação (-j ação). Antes de --dport, deve ser especificado um protocolo (-p). Exemplo:

```
-d 127.20.0.1 -p tcp --dport 80 -j DNAT --to 10.0.0.2
```

--sport --> assim como -s define um host de origem, --sport define uma porta de origem. Deve ser utilizado antes de uma ação (-j ação).

--to-port --> define uma porta de destino, após um REDIRECT.

Obs: A maioria dos dados básicos apresentados para a tabela filter continuam valendo. Exemplo: -p servirá para definir um protocolo de rede; -d define um host de destino.

Ações

SNAT --> Utilizado com POSTROUTING para fazer ações de mascaramento da origem.

DNAT --> Utilizado com PREROUTING e OUTPUT para fazer ações de redirecionamento de portas e servidores, balanceamento de carga e proxy transparente. Caso a porta de destino não seja especificada, valerá a porta de origem. No firewall, a porta que será redirecionada não pode existir ou estar ocupada por um daemon.

MASQUERADE --> Faz mascaramento na saída de dados.

REDIRECT --> Redireciona uma requisição para uma porta local do firewall.

Exemplos comentados de regras de firewall (tabela nat)

```
-----  
#iptables -t nat -L
```

Mostra as regras de NAT ativas.

```
-----  
#iptables -t nat -F
```

Apaga todas as regras de NAT existentes.

```
-----  
#iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE
```

Todos os pacotes que saírem pela interface ppp0 (modem) serão mascarados. Isso dá um nível de segurança elevado à rede que está atrás da ppp0. É uma boa regra para navegação na Internet. Note que esse tipo de mascaramento não usa SNAT.

```
-----  
#iptables -t nat -A POSTROUTING -d 0/0 -j MASQUERADE
```

Tem o mesmo efeito da regra anterior. No entanto, parece ser menos segura, pois estabelece que qualquer pacote destinado a qualquer outra rede, diferente da interna, será mascarado. A regra anterior refere-se aos pacotes que saem por determinada interface. A opção -d 0/0 poderia ser -d 0.0.0.0/0 também. É uma outra regra para navegação na Internet.

```
-----  
#iptables -t nat -A PREROUTING -t nat -p tcp -d 10.0.0.2 --dport 80 -j DNAT --to 172.20.0.1
```

Redireciona todos os pacotes destinados à porta 80 da máquina 10.0.0.2 para a máquina 172.20.0.1. Esse tipo de regra exige a especificação do protocolo. Como não foi especificada uma porta de destino, a porta de origem (80) será mantida como destino.

```
-----  
#iptables -t nat -A OUTPUT -p tcp -d 10.0.0.10 -j DNAT --to 10.0.0.1
```

Qualquer pacote TCP, originado na máquina firewall, destinado a qualquer porta da máquina 10.0.0.10, será desviado para a máquina 10.0.0.1 .

```
-----  
#iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to 200.20.0.1
```

Essa regra faz com que todos os pacotes que irão sair pela interface eth0 tenham o seu endereço de origem alterado para 200.20.0.1 .

```
-----  
#iptables -t nat -A PREROUTING -i eth0 -j DNAT --to 172.20.0.1
```

Todos os pacotes que entrarem pela eth0 serão enviados para a máquina 172.20.0.1 .

```
-----  
#iptables -t nat -A PREROUTING -i eth0 -j DNAT --to 172.20.0.1-172.20.0.3
```

Aqui haverá o load balance. Todos os pacotes que entrarem pela eth0 serão distribuídos entre as máquinas 172.20.0.1 , 172.20.0.2 e 172.20.0.3 .

```
-----  
#iptables -t nat -A PREROUTING -s 10.0.0.0/8 -p tcp --dport 80 -j REDIRECT --to-port 3128
```

Todos os pacotes TCP que vierem da rede 10.0.0.0, com máscara 255.0.0.0, destinados à porta 80 de qualquer host, não sairão; serão redirecionados para a porta 3128 do firewall. Isso é o passo necessário para fazer um proxy transparente. O proxy utilizado deverá aceitar esse tipo de recurso. No caso, o Squid, que aceita transparência, deverá estar instalado na máquina firewall, servindo na porta 3128.

```
-----  
#iptables -t nat -A POSTROUTING -s 192.168.1.0/24 -o eth1 -j SNAT 200.20.5.0/24
```

Uma situação interessante: todos os pacotes que saírem da rede 192.168.1.0 serão transformados em 200.20.5.0 .

```
-----
```

Execução do mascaramento destinado à Internet

Por ser uma atividade perigosa, o acesso à Internet deve ser feito com um máximo grau de segurança. Assim, vejamos as regras básicas para permitir uma navegação adequada.

Primeiro exemplo: uma rede na Internet

Vamos permitir que a rede 10.0.0.0 navegue na Internet. A máquina firewall será a 10.0.0.1. Regras:

```
#echo 1 > /proc/sys/net/ipv4/ip_forward
#modprobe iptable_nat
#iptables -t nat -A POSTROUTING -s 10.0.0.0/8 -o ppp0 -j MASQUERADE
```

O procedimento é totalmente seguro, pois discrimina uma origem, que só poderá sair pela ppp0, de forma mascarada.

Segundo exemplo: alguns hosts na Internet

Vamos permitir que alguns hosts, no caso, o 10.0.0.10, o 10.0.0.20 e o 10.5.2.41, naveguem na Internet. A máquina firewall será a 10.0.0.1. Regras:

```
#echo 1 > /proc/sys/net/ipv4/ip_forward
#modprobe iptable_nat
#iptables -t nat -A POSTROUTING -s 10.0.0.10 -o ppp0 -j MASQUERADE
#iptables -t nat -A POSTROUTING -s 10.0.0.20 -o ppp0 -j MASQUERADE
#iptables -t nat -A POSTROUTING -s 10.5.2.41 -o ppp0 -j MASQUERADE
```

Execução de FTP

Para executar sessões de FTP, será necessário o carregamento de dois módulos:

```
#insmod ip_conntrack_ftp
#insmod ip_nat_ftp
```

Salvando e recuperando regras

As regras de iptables devem ser salvas com o comando iptables-save e carregadas com iptables-restore. O roteamento estático e o carregamento dos módulos FTP devem ser feitos separadamente. Apenas para ilustrar, vamos executar o salvamento e a recuperação das regras.

```
--> Criando as regras
#iptables -t nat -A POSTROUTING -s 10.0.0.10 -o ppp0 -j MASQUERADE
#iptables -t nat -A POSTROUTING -s 10.0.0.20 -o ppp0 -j MASQUERADE
#iptables -t nat -A POSTROUTING -s 10.5.2.41 -o ppp0 -j MASQUERADE
```

```
--> Salvando
#iptables-save > /etc/firewall
```

--> Recuperação

Um script de recuperação, inicializado pelo /etc/rc.d/rc.local, poderia ter a seguinte estrutura:

```
#!/bin/bash
echo 1 > /proc/sys/net/ipv4/ip_forward
modprobe iptable_nat
iptables-restore < /etc/firewall
insmod ip_conntrack_ftp
insmod ip_nat_ftp
```

Aumentando o nível de segurança

Caso deseje aumentar o nível de segurança, evitando ataques diversos, digite COMO PRIMEIRAS regras:

```
#iptables -A FORWARD -m unclean -j DROP
```

```
#iptables -A FORWARD -p icmp --icmp-type echo-request -j DROP
```

```
#iptables -A FORWARD -p icmp --icmp-type echo-request -m limit --limit 1/s -j ACCEPT
```

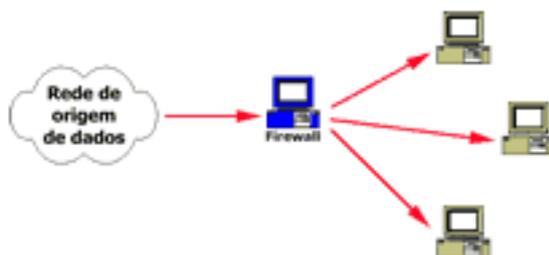
```
#iptables -A FORWARD -p tcp -m limit --limit 1/s -j ACCEPT
```

```
#iptables -A FORWARD -p tcp --tcp-flags SYN,ACK,FIN,RST -m limit --limit 1/s -j ACCEPT
```

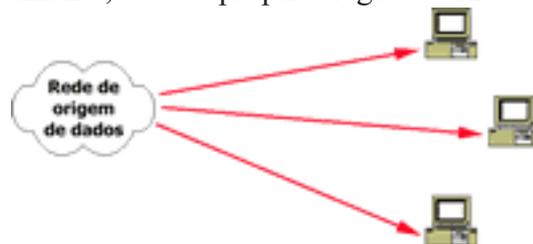
Topologias de firewall

O posicionamento de um firewall dentro da rede é de extrema importância para ela. Há duas possibilidades básicas para a utilização de um firewall: firewall isolado e firewall incorporado, sendo este último o mais inseguro e menos desejável.

O firewall será isolado quando estiver entre máquinas, com função exclusiva de firewall:



O firewall será incorporado quando não houver uma máquina isolada como firewall. Nesse caso, as máquinas da rede deverão estabelecer, individualmente, as suas próprias regras de firewall. É o sistema mais inseguro:



Nada impede que os dois sistemas sejam utilizados em parceria. Cabe ressaltar que no firewall incorporado há maior nível de insegurança, uma vez que outros processos rodam junto com o firewall.

4. Segurança do sistema de firewall

O sistema de firewall deve ser protegido para que o restante da rede também tenha segurança. Assim, algumas regras básicas devem ser observadas:

--> Feche a máquina firewall, de modo que todas os pacotes destinados diretamente a ela sejam descartados:

```
#iptables -P INPUT DROP
```

Em seguida, aos poucos, abra o que for necessário.

--> Prefira topologia de firewall isolado combinado com firewall incorporado;

--> Atualize sempre o firewall e o kernel;

--> NUNCA rode qualquer serviço, principalmente os remotos, como telnet e ftp, na máquina firewall, quando se tratar de firewall isolado;

--> Se tiver que administrar remotamente um firewall, utilize ssh;

--> Nunca cadastre qualquer usuário na máquina firewall, caso se trate de firewall isolado;

--> Utilize TCP Wrappers totalmente fechado (ALL:ALL em /etc/hosts.deny) na máquina de firewall isolado; abra o ssh (em /etc/hosts.allow) apenas para os clientes que forem fazer administração remota;

--> Anule as respostas a ICMP 8 (echo reply) no firewall isolado, para evitar identificação da topologia na rede e ataques de Ping of Death. A melhor forma de se fazer isso é atuando sobre regras do kernel, com o comando:

```
#echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_all
```

--> Não insira referências ao firewall no DNS;

--> Não deixe o firewall isolado com cara de firewall. Dê um nome descaracterizado para ele;

--> Faça log de ações suspeitas que estiverem ocorrendo na rede;

--> Teste, teste, teste novamente.

LINKS - RELATIVOS

<http://netfilter.samba.org>

<http://netfilter.filewatcher.org>



Arquivo elaborado por LinuxClube.com
<http://www.linuxclube.com>