Sobre a licença

creative commons

Atribuição-Compartilhamento pela mesma Licença 2.5 Brasil

Você pode:

- copiar, distribuir, exibir e executar a obra
- criar obras derivadas

Free Cultural APPROVED FOR Works

Sob as seguintes condições:

- **Atribuição**. Você deve dar crédito ao autor original, da forma especificada pelo autor ou licenciante.
- **Compartilhamento pela mesma Licença**. Se você alterar, transformar, ou criar outra obra com base nesta, você somente poderá distribuir a obra resultante sob uma licença idêntica a esta.

- Para cada novo uso ou distribuição, você deve deixar claro para outros os termos da licença desta obra.
- Qualquer uma destas condições podem ser renunciadas, desde que Você obtenha permissão do autor.
- Nothing in this license impairs or restricts the author's moral rights.

Para cada novo uso ou distribuição, você deve deixar claro para outros os termos da licença desta obra.
No caso de criação de obras derivadas, os logotipos do CGI.br, NIC.br, IPv6.br e CEPTRO.br não devem ser utilizados.
Na atribuição de autoria, essa obra deve ser citada da seguinte forma:
Apostila “Curso IPv6 básico” do NIC.br, disponível no sítio http://curso.ipv6.br ou através do e-mail ipv6@nic.br.
Qualquer uma destas condições podem ser renunciadas, desde que você obtenha permissão do autor.
Se necessário, o NIC.br pode ser consultado através do email ipv6@nic.br.
Nada nesta licença prejudica ou restringe os direitos morais do autor.

Publicação
ceptro.br

IPv6.br

Curso IPv6 básico

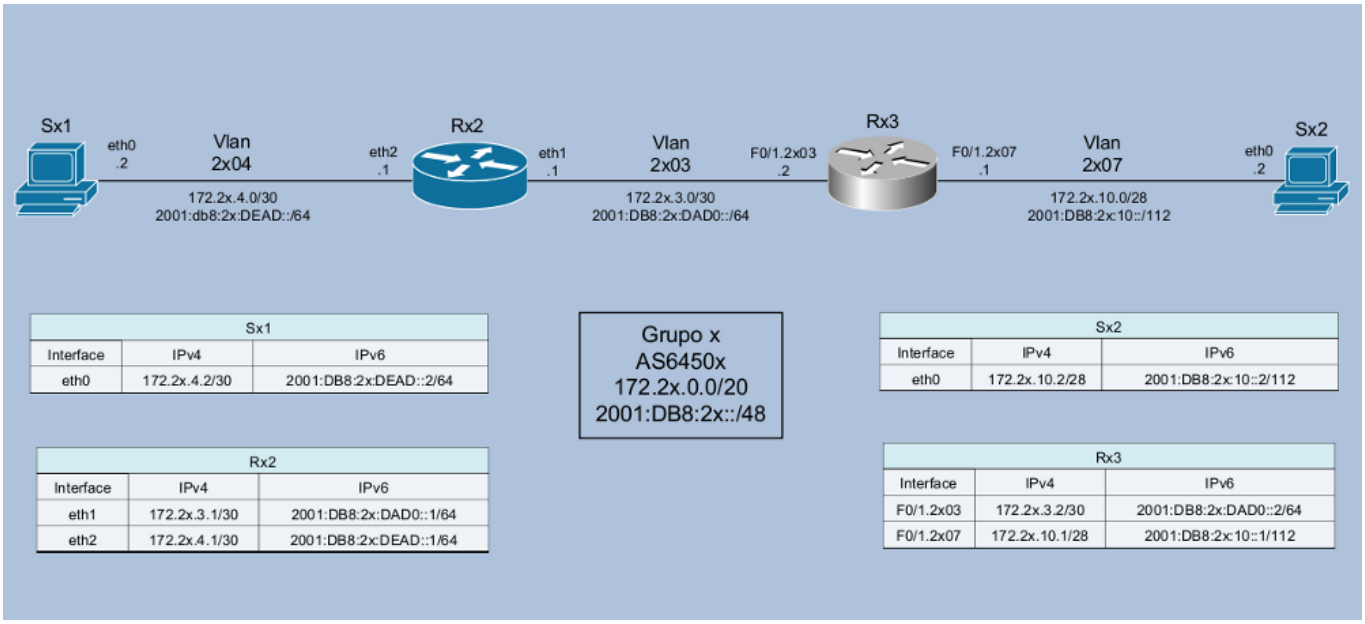
Laboratório: Firewall IPv6

cgi.br nic.br

Laboratório – Firewall IPv6

Objetivo: Implementar um firewall simples nos servidores do AS, com suporte nativo a IPv6, utilizando ip6tables.

Cenário inicial: Cada AS possui acesso a um roteador Cisco, um roteador Linux/Quagga, e dois servidores Linux. Não há políticas de roteamento externo ou protocolo de roteamento interno (IGP) implementados, nem para IPv4 nem IPv6. Há apenas as configurações de endereçamento estático IPv4 e IPv6. O grupo deve testar a comunicação dentro do próprio AS (use mtr, ping e traceroute IPv4 e IPv6, por exemplo).



Sx1		
Interface	IPv4	IPv6
eth0	172.2x.4.2/30	2001:DB8:2x:DEAD::2/64

Sx2		
Interface	IPv4	IPv6
eth0	172.2x.10.2/28	2001:DB8:2x:10::2/112

Rx2		
Interface	IPv4	IPv6
eth1	172.2x.3.1/30	2001:DB8:2x:DAD0::1/64
eth2	172.2x.4.1/30	2001:DB8:2x:DEAD::1/64

Rx3		
Interface	IPv4	IPv6
F0/1.2x03	172.2x.3.2/30	2001:DB8:2x:DAD0::2/64
F0/1.2x07	172.2x.10.1/28	2001:DB8:2x:10::1/112

Exercício 1 - Configurando ip6tables.

É preciso ter uma atenção maior na utilização de firewall's em redes IPv6, visto que, ao contrário da maioria das redes IPv4, a rede interna não é mais “protegida” pela utilização de endereços IP privados (RFC 1918). Com a adoção do protocolo IPv6 todos os hosts podem utilizar endereços válidos com conectividade direta a Internet e alcance a todos os hosts da rede interna que tenham IPv6 habilitado.

Vamos utilizar o seguinte script com regras para o ip6tables.

```
#!/bin/sh

PATH=/sbin:/bin:/usr/sbin:/usr/bin

# caminho do iptables
iptables="/sbin/ip6tables"

# Meus IPs
# Acrescentar os IPs v6 do servidor aqui
ips_locais="2001:DB8:XX:DEAD::2/128 FE80::XXXX:XXFF:FEXX:XXXX/128 FF02::1:FF00:0/104
FF02::1/128"

start () {
        echo "Iniciando o filtro de pacotes: ip6tables..."

        # A politica padrao eh recusar todos os pacotes
        echo "Configurando a politica padrao para recusar todos os pacotes"
        $iptables -F
        $iptables -P INPUT DROP
        $iptables -P OUTPUT DROP
        $iptables -P FORWARD DROP

        # Permitir trafego ilimitado para o localhost
        echo "Permitindo trafego ilimitado para o localhost"
        $iptables -A INPUT  -i lo -j ACCEPT
        $iptables -A OUTPUT -o lo -j ACCEPT

        # Conexoes permitidas de entrada e saida para este servidor
        for ip in $ips_locais
          do
            echo -n "Permitindo algumas conexoes de entrada para o este servidor (IP $ip)..."

            # Abrindo o ssh para todos
            echo -n "ssh "
            $iptables -A INPUT -p tcp -s ::/0 --sport 513:65535 -d $ip --dport 22 -j ACCEPT
            $iptables -A OUTPUT -p tcp -d ::/0 --dport 513:65535 -s $ip --sport 22 -j ACCEPT

            # Trafego HTTP
            echo -n "http "
            $iptables -A INPUT -p tcp -d $ip --dport 80 -j ACCEPT
            $iptables -A OUTPUT -p tcp -s $ip --sport 80 -j ACCEPT

            # Permitindo Traceroute
            $iptables -A INPUT -p udp --dport 33434:65535 -d $ip -j ACCEPT
            $iptables -A OUTPUT -p udp --dport 33434:65535 -s $ip -j ACCEPT

            # Permitindo o envio de mensagens ICMPv6
            echo -n "icmp out "
            $iptables -A OUTPUT -p icmpv6 -s $ip -j ACCEPT

            ###### RFC 4890 #####
            ###### Trafego ICMPv6 que NAO DEVE ser DESCARTADO ######
            echo -n "icmp in "
            # ECHO REQUESTS E RESPONSES (Type 128 e 129)
            # ========================
            $iptables -A INPUT -p icmpv6 --icmpv6-type echo-request -d $ip -j ACCEPT
```

```
            $iptables -A INPUT -p icmpv6 --icmpv6-type echo-reply -d $ip -j ACCEPT

            # DESTINATION UNREACHABLE (Type 1)
            # ========================
            $iptables -A INPUT -p icmpv6 --icmpv6-type destination-unreachable -d $ip -j
ACCEPT

            # PACKET TOO BIG (Type 2)
            # ===============
            $iptables -A INPUT -p icmpv6 --icmpv6-type packet-too-big -d $ip -j ACCEPT

            # TIME EXCEEDED (Type 3)
            # ==============
            $iptables -A INPUT -p icmpv6 --icmpv6-type ttl-zero-during-transit -d $ip -j
ACCEPT
            $iptables -A INPUT -p icmpv6 --icmpv6-type ttl-zero-during-reassembly -d $ip -j
ACCEPT

            # PARAMETER PROBLEM (Type 4)
            # ==================
            $iptables -A INPUT -p icmpv6 --icmpv6-type unknown-option -d $ip -j ACCEPT
            $iptables -A INPUT -p icmpv6 --icmpv6-type unknown-header-type -d $ip -j ACCEPT
            $iptables -A INPUT -p icmpv6 --icmpv6-type bad-header -d $ip -j ACCEPT

            # NEIGHBOR DISCOVERY
            # ==================
            # RS (Type 133)
            $iptables -A INPUT -p icmpv6 --icmpv6-type 133 -d $ip -j ACCEPT
            # RA (Type 134)
            $iptables -A INPUT -p icmpv6 --icmpv6-type 134 -d $ip -j ACCEPT
            # NS (Type 135)
            $iptables -A INPUT -p icmpv6 --icmpv6-type 135 -d $ip -j ACCEPT
            # NA (Type 136)
            $iptables -A INPUT -p icmpv6 --icmpv6-type 136 -d $ip -j ACCEPT
            # Inverse Neighbor Discovery Solicitation (Type 141)
            $iptables -A INPUT -p icmpv6 --icmpv6-type 141 -d $ip -j ACCEPT
            # Inverse Neighbor Discovery Advertisement (Type 142)
            $iptables -A INPUT -p icmpv6 --icmpv6-type 142 -d $ip -j ACCEPT

            # MLD
            # ===
            # Listener Query (Type 130)
            $iptables -A INPUT -p icmpv6 --icmpv6-type 130 -d $ip -j ACCEPT
            # Listener Report (Type 131)
            $iptables -A INPUT -p icmpv6 --icmpv6-type 131 -d $ip -j ACCEPT
            # Listener Done (Type 132)
            $iptables -A INPUT -p icmpv6 --icmpv6-type 132 -d $ip -j ACCEPT
            # Listener Report v2 (Type 143)
            $iptables -A INPUT -p icmpv6 --icmpv6-type 143 -d $ip -j ACCEPT

            # SEND
            # ====
            # Certificate Path Solicitation (Type 148)
            $iptables -A INPUT -p icmpv6 --icmpv6-type 148 -d $ip -j ACCEPT
            # Certificate Path Advertisement (Type 149)
            $iptables -A INPUT -p icmpv6 --icmpv6-type 149 -d $ip -j ACCEPT

            # Multicast Router Discovery
            # ==========================
            # Multicast Router Advertisement (Type 151)
            $iptables -A INPUT -p icmpv6 --icmpv6-type 151 -d $ip -j ACCEPT
            # Multicast Router Solicitation (Type 152)
            $iptables -A INPUT -p icmpv6 --icmpv6-type 152 -d $ip -j ACCEPT
            # Multicast Router Termination (Type 153)
            $iptables -A INPUT -p icmpv6 --icmpv6-type 153 -d $ip -j ACCEPT

            ##### Trafego ICMPv6 que NORMALMENTE NAO DEVE ser DESCARTADO #####
            # Mobilidade IPv6 ### Apenas as habilite se o noh for um Noh Movel ###
            # ===============
```

```
            # Home Agent Address Discovery Request (Type 144)
            # $iptables -A INPUT -p icmpv6 --icmpv6-type 144 -d $ip -j ACCEPT
            # Home Agent Address Discovery Reply (Type 145)
            # $iptables -A INPUT -p icmpv6 --icmpv6-type 145 -d $ip -j ACCEPT
            # Mobile Prefix Solicitation (Type 146)
            # $iptables -A INPUT -p icmpv6 --icmpv6-type 146 -d $ip -j ACCEPT
            # Mobile Prefix Advertisement (Type 147)
            # $iptables -A INPUT -p icmpv6 --icmpv6-type 147 -d $ip -j ACCEPT

            ###### Casos especificos ######
            ## Algumas mensagens não precisam de tratamento:
            #  - Router Renumbering (Type 138): Devem ser autenticadas com IPSec
            #
            #
            ## Algumas mensagens precisam de politicas especificas:
            #  - Redirect (Type 137): Podem oferecer riscos a segurança. Sua
            # utilização deve ser estudada caso a caso.
            #
            #
            ## Mensagens ainda nao definidas pela a IANA ou de uso experimental
            # devem ser sempre descartadas.
            ## A nao ser que exista um caso muito especifico na rede e que elas
            # sejam utilizadas.
           echo .

        done

        # Descartando tudo mais
        echo "Descartando todos os demais pacotes... "
        $iptables -A INPUT -s ::/0 -j DROP
        $iptables -A OUTPUT -d ::/0 -j DROP
}

stop () {
        echo "Parando o filtro de pacotes: ip6tables..."
        $iptables -P INPUT ACCEPT
        $iptables -F INPUT
        $iptables -P OUTPUT ACCEPT
        $iptables -F OUTPUT
        $iptables -P FORWARD ACCEPT
        $iptables -F FORWARD
        $iptables -F LOGDROP
        $iptables -X LOGDROP
        echo "Todas as regras e cadeias estao limpas."
        echo "Tome cuidado... Isso eh perigoso!!"
        echo "Execute: ** /etc/init.d/ip6tables start ** assim que possivel."
}

status () {
       $iptables --list -v
}
case "$1" in
    start)
        start
    ;;
    stop)
        stop
    ;;
    try|test)
        start
        sleep 10
        stop
    ;;

    restart|reload|force-reload)
        stop
        sleep 2
        start
    ;;
```

```
    status)
        status
    ;;
    *)
        echo "Uso: /etc/init.d/ip6tables {start|stop|restart|status|try}" >&2
        exit 1
    ;;
esac

exit
```

Faça o download desse script no endereço

http://[xxxx:xxxx:x:xxxx::xxx]/ipt6tables.txt

Basicamente, as regras de firewall aqui aplicadas tem como política padrão descartar todos os tipos de pacotes, permitindo apenas o acesso ao nó via ssh na porta 22, acesso via http na porta 80, a rastreabilidade do nó via traceroute e a utilização das mensagens ICMPv6 de acordo com as recomendações da RFC 4890.

Agora vamos aplicar essas regras em um dos servidores do AS. No servidor Sx1, crie o arquivo ip6tables no diretório /etc/init.d/ e adicione o conteúdo do script baixado anteriormente.

```
[root@SX1 /]# cd /etc/init.d/
[root@SX1 /]# cat -> ip6tables
[Ctrl+V]
[Ctrl+D]
```

Altere os endereços contidos na linha 10 do arquivo de acordo com a numeração do servidor, salve o arquivo e reinicie o serviço do ip6tables:

```
[root@SX1 /]# /etc/init.d/ip6tables restart
```

Exercício 2 - Testando as regras do firewall

Seu firewall IPv6 já deve estar funcionando normalmente. Agora vamos realizar alguns testes para analisarmos sua configuração e se há diferenças na definição das regras para o iptables (IPv4) e o ip6tables (IPv6).

A partir do servidor Sx2, acesse o servidor Sx1 via ssh:

```
[root@SX2 /]# ssh root@2001:DB8:2X:DEAD::2
The authenticity of host '2001:db8:2X:dead::2 (2001:db8:2X:dead::2)' can't be
established.
RSA key fingerprint is f9:66:86:4b:d6:81:1b:c7:79:27:1f:54:76:00:ba:d9.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '2001:db8:2X:dead::2' (RSA) to the list of known hosts.
root@2001:db8:2X:dead::2's password:
```

Acesse também o serviço de http do sevidor Sx1 utilizando o *browser* elinks:

```
[root@SX2 /]# elinks [2001:DB8:2X:DEAD::2]
```

Após a realização dos dois acessos, altere o script do ip6tables no servidor Sx1 para bloquear esses dois serviços, comentando as linhas correspondentes:

```
...
# Abrindo o ssh para todos
#echo -n "ssh "
#$iptables -A INPUT -p tcp -s ::/0 --sport 513:65535 -d $ip --dport 22 -j ACCEPT
#$iptables -A OUTPUT -p tcp -d ::/0 --dport 513:65535 -s $ip --sport 22 -j ACCEPT


# Trafego HTTP
#echo -n "http "
#$iptables -A INPUT -p tcp -d $ip --dport 80 -j ACCEPT
#$iptables -A OUTPUT -p tcp -s $ip --sport 80 -j ACCEPT
...
```

Reinicie o serviço do ip6tables e tente realizar o acesso a esses serviços novamente, a partir do servidor Sx2.

```
[root@SX1 /]# /etc/init.d/ip6tables restart
```

Também testaremos a utilização do comando traceroute6 traçando a rota a partir do servidor Sx2 até o servidor Sx1:

```
[root@SX2 /]# traceroute6 2001:db8:2X:DEAD::2
traceroute to 2001:db8:2X:DEAD::2 (2001:db8:2X:dead::2) from 2001:db8:2X:10::2,
      30 hops max, 24 byte packets
 1  2001:db8:2X:10::1 (2001:db8:2X:10::1)  0.745 ms  0.597 ms  0.624 ms
 2  2001:db8:2X:dad0::1 (2001:db8:2X:dad0::1)  0.43 ms  0.411 ms  0.323 ms
 3  2001:db8:2X:dead::2 (2001:db8:2X:dead::2)  1.468 ms  0.4 ms  0.337 ms
```

Altere novamente o script do ip6tables no Sx1, comentando as linhas que permitem a utilização do comando traceroute:

```
...
# Permitindo Traceroute
#$iptables -A INPUT -p udp --dport 33434:65535 -d $ip -j ACCEPT
#$iptables -A OUTPUT -p udp --dport 33434:65535 -s $ip -j ACCEPT
...
```

Reinicie o serviço do ip6tables e tente traçar novamente a rota para o servidor Sx1 a partir do servidor Sx2.

Também podemos testar a conectividade IPv6 entre os dois servidores do As, através de pings de um servidor para ou outro.

```
[root@SX1 /]# ping6 2001:db8:2X:10::1
PING 2001:db8:2X:10::1(2001:db8:2X:10::1) 56 data bytes
64 bytes from 2001:db8:2X:10::1: icmp_seq=0 ttl=63 time=0.658 ms
64 bytes from 2001:db8:2X:10::1: icmp_seq=1 ttl=63 time=0.647 ms
64 bytes from 2001:db8:2X:10::1: icmp_seq=2 ttl=63 time=0.659 ms
...


[root@SX2 /]# ping6 2001:db8:2X:dead::2
PING 2001:db8:2X:dead::2(2001:db8:2X:dead::2) 56 data bytes
64 bytes from 2001:db8:2X:dead::2: icmp_seq=0 ttl=62 time=1.61 ms
64 bytes from 2001:db8:2X:dead::2: icmp_seq=1 ttl=62 time=0.427 ms
64 bytes from 2001:db8:2X:dead::2: icmp_seq=2 ttl=62 time=0.431 ms
...
```

Vamos agora alterar o script do ip6tables do servidor Sx1, comentando as linhas que permitem o recebimento de mensagens ICMPv6 echo-request e echo-reply:

```
# ECHO REQUESTS E RESPONSES (Type 128 e 129)
# ========================
#$iptables -A INPUT -p icmpv6 --icmpv6-type echo-request -d $ip -j ACCEPT
#$iptables -A INPUT -p icmpv6 --icmpv6-type echo-reply -d $ip -j ACCEPT
```

Reinicie o serviço do ip6tables e realize novamente os testes de conectividade entre os servidores Sx1 e Sx2.

A RFC 4890 recomenda que não se bloqueie a utilização de pings, dizendo que está prática de segurança é desnecessária, visto que a realização de varredura de endereços em uma rede IPv6 é praticamente impossível. O que você acha desta recomendação? O bloqueio de pings é importante ou não?

Exercício 3 – Bloqueando mensagens ICMPv6

Vamos testar agora as regras que permitem o envio e o recebimento das mensagens ICMPv6 utilizadas pelo protocolo de Descoberta de Vizinhança.

Primeiro, habilite o serviço radvd no roteador Rx2, editando ou criando o arquivo /etc/radvd.conf com o seguinte conteúdo:

- No roteador Rx2:

```
 interface eth2 {
       AdvSendAdvert on;
       MinRtrAdvInterval 3;
       MaxRtrAdvInterval 30;
       AdvLinkMTU 1500;
       prefix 2001:DB8:2X:DEAD::/64 {
               AdvOnLink on;
               AdvAutonomous on;
               AdvRouterAddr on;
               AdvPreferredLifetime 90;
               AdvValidLifetime 120;
       };
  };
```

Inicie o Radvd

- No roteador Rx2:

```
[root@RX2 /]# /etc/init.d/radvd start
```

Caso ocorra algum erro ao se iniciar o processo do Radvd, verifique o arquivo de logs do roteador Rx2:

```
[root@RX2 /]#tail /var/log/messages
```

Verifique se o servidor Sx1 recebeu um endereço IPv6 Unicast Global através do mecanismo de autoconfiguração stateless.

Agora, vamos comentar as linhas que permitem o recebimento das mensagens RA, RS, NA e NS:

```
# NEIGHBOR DISCOVERY
# ==================
# RS (Type 133)
#$iptables -A INPUT -p icmpv6 --icmpv6-type 133 -d $ip -j ACCEPT
# RA (Type 134)
#$iptables -A INPUT -p icmpv6 --icmpv6-type 134 -d $ip -j ACCEPT
# NS (Type 135)
#$iptables -A INPUT -p icmpv6 --icmpv6-type 135 -d $ip -j ACCEPT
# NA (Type 136)
#$iptables -A INPUT -p icmpv6 --icmpv6-type 136 -d $ip -j ACCEPT
```

Reinicie o serviço do ip6tables e verifique se o servidor Sx1 continua recebendo um endereço via autoconfiguração stateless. Note que o endereço atribuído anteriormente pode demorar alguns minutos para deixar de ser utilizado pela interface.

Agora descomente apenas as linhas necessárias para que esse serviço volte a funcionar normalmente. Quais mensagens são necessárias para que a autoconfiguração stateless funcione?

Também é possível melhorar esse script permitindo que os endereços FF02::1:FF00:0/104 e FF02::1/128 recebam apenas as mensagens que realmente são destinadas a eles. Consulte na apostila teórica quais são essas mensagens e faça as alterações necessárias no script. Em seguida teste para verificar se a autoconfiguração stateless continua funcionado normalmente.