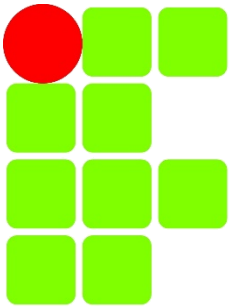

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SUL DE MINAS GERAIS
Campus Inconfidentes

LEONARDO PEDROSO ALVARENGA
RENAN ALEXANDRE MACIEL

FIREWALL EM REDES IPv6: COMPARAÇÃO ENTRE OPNSENSE E IPTABLES

INCONFIDENTES-MG
2017

LEONARDO PEDROSO ALVARENGA
RENAN ALEXANDRE MACIEL

FIREWALL EM REDES IPv6: COMPARAÇÃO ENTRE OPNSENSE E IPTABLES

Trabalho de Conclusão de Curso apresentado como pré-requisito de conclusão do curso de Graduação em Tecnologia em Redes de Computadores no Instituto Federal de Educação, Ciência e Tecnologia do Sul de Minas Gerais – Campus Inconfidentes, para obtenção do título de Tecnólogo em Redes de Computadores.

Orientador: Vinicius Ferreira de Souza

INCONFIDENTES-MG
2017

LEONARDO PEDROSO ALVARENGA
RENAN ALEXANDRE MACIEL

FIREWALL EM REDES IPv6: COMPARAÇÃO ENTRE OPNSENSE E IPTABLES

Data de aprovação: 31 de Outubro de 2017

Vinícius Ferreira de Souza – IFULDEMINAS – Inconfidentes

Kleber Marcelo da Silva Rezende – IFULDEMINAS – Inconfidentes

---

Igor Oliveira Lara – IFULDEMINAS – Inconfidentes

Instituto Federal de Educação, Ciência e Tecnologia
Sul de Minas Gerais
Campus Inconfidentes
Praça Tiradentes, 416 - Centro - CEP 37576-000
Telefone: (35) 3464-1200

FIREWALL EM REDES IPv6: Comparação entre OPNsense e Iptables

Vinícius F. de SOUZA, Leonardo P. ALVARENGA, Renan A. MACIEL

Setor de Informática e Redes
IFSULDEMINAS – Campus Inconfidentes – Inconfidentes, MG – Brasil

vinicius.souza@ifsuldeminas.edu.br, leonardokbj@gmail.com,
renan_alle@hotmail.com

Abstract. *The evolution of the Internet and the growing demand for IP addresses caused a shortage of IPv4 addresses and motivated the emergence of a new generation of the IP protocol, called IPv6. However, the transition from IPv4 to IPv6 poses many challenges for the network administrator, and one of the main concerns refers to security. This work presents a comparative analysis between two tools used to configure a firewall in IPv6 networks, in order to verify which one is more flexible and secure for this purpose. Through the configuration of a security policy and through the verification of some features of the IPv6 protocol, it was possible to verify some advantages and disadvantages in the use of each of the software analyzed: OPNsense and Iptables. The results pointed to Iptables as a better solution for the configuration of firewalls in IPv6 environments.*

Resumo. *A evolução da Internet e a crescente demanda por endereços IP, ocasionaram uma escassez de endereços IPv4 e motivaram o surgimento de uma nova geração do protocolo IP, denominada IPv6. No entanto, a transição do IPv4 para o IPv6 impõe muitos desafios para o administrador da rede, e uma das principais preocupações diz respeito à segurança. Este trabalho apresenta uma análise comparativa entre duas ferramentas utilizadas para a configuração de um firewall em redes IPv6, com o objetivo de verificar qual delas é mais flexível e segura para esse propósito. Através da configuração de uma política de segurança e mediante a verificação de alguns recursos do protocolo IPv6, foi possível constatar algumas vantagens e desvantagens no uso de cada um dos softwares analisados: OPNsense e Iptables. Os resultados apontaram o Iptables como uma solução mais adequada para a configuração de firewalls em ambientes IPv6.*

1. Introdução

O protocolo IPv6 foi concebido com a proposta de substituir totalmente, a longo prazo, o IPv4 como protocolo de endereçamento em redes de computadores. O IPv6 apresenta uma série de vantagens em relação ao seu antecessor e destaca-se, principalmente, por disponibilizar um maior espaço de endereçamento que permite atender toda a demanda atual da Internet, ao

contrário do IPv4, sendo essa a maior motivação para o desenvolvimento do mesmo. Desde o seu lançamento oficial, em 6 de junho de 2012, vem sendo implantado gradativamente, funcionando junto com o IPv4 numa situação que pode ser denominada de pilha dupla ou *dual stack* (BRITO, 2013).

O IPv6, também é acompanhado do ICMPv6, e este por sua vez possui suas variantes denominadas "*types*", assim como o ICMPv4. Porém a função desses *types* no funcionamento do protocolo de endereçamento é completamente diferente do antecessor, criando um sistema onde os dois protocolos (IPv6 e ICMPv6) precisam trabalhar juntos para o funcionamento da camada de rede. Dessa forma, mostrou-se necessário o estudo do funcionamento destes protocolos e também o uso das ferramentas corretas para proteger a rede de novos ataques que exploram este novo sistema (ARJUMAN, 2013).

O IPv6 também foi simplificado em relação ao seu antecessor, a fim de reduzir o custo de processamento e limitar a largura de banda do cabeçalho IPv6. No entanto, ele também traz mudanças na maneira como as opções do cabeçalho IP são codificadas para permitir um encaminhamento mais eficiente, e maior flexibilidade para a introdução de novas opções no futuro. No IPv6, a informação opcional da camada de internet é codificada em cabeçalhos separados que podem ser colocados entre o cabeçalho IPv6 e o cabeçalho da camada superior em um pacote. Um pacote IPv6 pode carregar cabeçalhos de extensão zero, um ou mais, cada um identificado pelo campo "Próximo Cabeçalho" do cabeçalho anterior. Este novo modo de codificar informações opcionais também trouxe uma série de vulnerabilidades e complicações que vêm sendo analisadas desde o lançamento do protocolo. O mecanismo de fragmentação, por exemplo, utiliza os cabeçalhos de extensão e continua tendo problemas no IPv6, assim como no IPv4. No caso do IPv6, ela traz complicações em cenários onde os dispositivos e nós não obedecem às normas reguladoras do protocolo (ATLASIS, 2012).

Com a popularização do IPv6, e a maior adesão por parte dos usuários da Internet em geral, tornou-se necessária uma maior atenção com a questão da segurança, tendo em vista que o novo protocolo traz grandes alterações em relação ao seu antecessor como dito anteriormente e, consequentemente, demanda novas medidas e conhecimento específico para garantir a segurança da rede (NIC.BR, 2016).

Quando o assunto é segurança, a presença de um *Firewall* para analisar o tráfego da rede torna-se imprescindível. Embora existam várias soluções *Open Source* baseadas em *FreeBSD* ou *Linux*, a maioria delas não possui suporte ao IPv6 (NIC.BR, 2016). Nesse contexto, foco deste trabalho, estão incluídos os softwares OPNsense, PfSense e Iptables. Também existem algumas soluções proprietárias com a mesma finalidade, tais como o *Cisco PIX* e o *MikroTik*.

Para esta análise comparativa foram selecionadas duas ferramentas *Open Source*, sendo uma baseada em *FreeBSD* (OPNsense) e a outra em *Linux* (Iptables). O OPNsense foi selecionado, em detrimento do PfSense que também é baseado em *FreeBSD*, por ser um projeto mais recente e que oferece várias atualizações de segurança. Também foi selecionado uma ferramenta chamada Chiron, que consegue manipular pacotes IPv6, possibilitando uma variedade de testes nos Firewalls.

O propósito deste trabalho é identificar algumas vantagens e desvantagens no uso de cada software analisado e verificar qual deles é mais flexível e seguro para a configuração de firewalls em redes IPv6. Os resultados dos testes realizados indicaram o Iptables como a solução mais adequada para tal finalidade.

2. Material e Métodos

Para a realização da comparação proposta, foram utilizadas as seguintes versões de softwares: OPNsense 17.1.4 e Iptables 1.4.21 com o sistema operacional Debian 3.16.43-2.

2.1 OPNsense

O OPNsense é uma plataforma que executa funções de roteamento e firewall. Surgiu como um *fork* do PfSense e do *m0n0wall* em 2014, e foi lançado oficialmente em janeiro de 2015. A sua interface gráfica é amigável e intuitiva, e a integração com o código é muito fluida. Possui muitas das funções presentes em firewalls comerciais e oferece sistemas de monitoramento e controle de tráfego completos (Figura 1).

O OPNsense oferece atualizações de segurança semanais com pequenos incrementos para reagir as novas ameaças emergentes de forma proativa. O conjunto de recursos do OPNsense inclui recursos como proxy, cache, modelagem de tráfego, detecção de intrusão e configuração fácil do cliente OpenVPN. A última versão é baseada no FreeBSD 11 para suporte a longo prazo e usa um framework MVC recentemente desenvolvido baseado em Phalcon. O foco da OPNsense na segurança traz recursos únicos, como a opção de usar o LibreSSL ao invés do OpenSSL (selecionável na GUI) e uma versão personalizada com base em HardenedBSD. Outra coisa interessante a se ressaltar, é o sistema de backup que é muito prático e funcional (OPNSENSE®, 2015).

Figura 1. Interface do OPNsense. Fonte: opnsense.org.

2.2 Iptables

O Iptables, nativo das distribuições *Linux* a partir do Kernel 2.4, proporciona flexibilidade ao permitir alterar, acrescentar ou retirar alguma regra a qualquer momento. Por outro lado, apresenta um certo grau de complexidade pelo fato de sua configuração ser realizada totalmente por linhas de comandos, sem o uso de qualquer tipo de interface gráfica (NETO, 2004).

As regras aplicadas, agem como filtros, que determinam quais datagramas do tráfego da rede serão processados ou descartados. Para isso é necessário configurar os filtros de pacotes, pelo tipo de protocolo, tipo de datagramas e endereços de origem e destino. Essas regras são armazenadas em *chains* e processadas na ordem que são inseridas, como ficam armazenadas no *kernel* são perdidas quando o sistema é reinicializado. Como solução deste problema um script com as regras de configuração deve ser criado dentro do diretório "/etc/init.d/", exemplificado abaixo (NETO, 2004).

Comando para criar o script:

#nano /etc/init.d/meuscript

Comando para dar permissão de execução:

#chmod 755 /etc/init.d/meuscript

Para finalizar, o comando para inicializar o script junto com o sistema:

 # update-rc.d meuscript defaults

O Linux utiliza-se de um recurso independente em termos de kernel para controlar e monitorar todo o tipo de fluxo de dados dentro de sua estrutura operacional. Mas não o faz sozinho, pois, sua função deve ser a de trabalhar ao lado de processos e tarefas tão somente. Para que o Kernel possa controlar seu próprio fluxo interno lhe fora agregada uma ferramenta batizada de Netfilter, ao qual, divide-se em três tabelas: Filter, NAT e Mangle (NETO, 2004).

2.2.1 Tabela Filter

A tabela padrão do Netfilter trata as situações implementadas num Firewall, onde um *datagrama* será avaliado por regras de firewall para liberar ou negar o seu fluxo, ou seja, é a tabela onde o "pente fino" nos pacotes é feito, decidindo que política adotar na passagem dos pacotes (NETO, 2004).

Nesta tabela, são inseridos os conjuntos de regras com finalidades gerais, desde que não alterem as configurações dos pacotes. A tabela filter possui três conjuntos de regras, também denominadas Chains (NETO, 2004):

- **INPUT:** Chain responsável pelas de regras aplicadas aos pacotes na entrada da interface, sendo assim, apenas os pacotes destinados ao IP da máquina atual serão avaliados por eventuais regras existentes nesta tabela. Onde trata os pacotes que dão entrada no servidor e que tem como destino o próprio servidor, porém, ela não trata pacotes que passam pelo firewall com destino à outra rede;
- **OUTPUT:** Chain responsável pelo filtro de regras dos pacotes originados por processos locais da máquina, ou seja, pacotes que tem origem no firewall e destino não firewall;
- **FORWARD:** Nesta Chain o filtro avalia os pacotes que estão sendo repassados pelo firewall e, não tem ele como destino ela e nem são originados por ele, ou seja, os pacotes de dados estão apenas passando pelo firewall.

2.2.2 Tabela NAT

Tabela responsável pelas implementações de NAT (*Network Address Translation*), ou seja, realizar operações de tradução sobre IP e/ou porta, tanto de origem como de destino, mas também pode optar pela recusa dos pacotes (NETO, 2004).

2.2.3 Tabela Mangle

Tabela que permite realizar alterações mais profunda e complexa nos pacotes, como alterar o TTL, TOS, etc., ou seja, alterações com um nível de complexidade maior (NETO, 2004).

2.2.4 SINTAXE

Como em toda linha de comando, o Iptables também se utiliza de uma sintaxe estruturada, abordada abaixo:

#iptables -t [tabela] <ordem> <chain> [condições] -j <ação>

Através dessas sintaxes, são criadas as regras que definirão as ações, ou seja, para liberar, bloquear ou rejeitar os pacotes de dados de trafego. Na Tabela 1 são apresentadas as ações na tabela Filter (NETFILTER.ORG).

Tabela 1. Ações na tabela FILTER.

Alvo	Opções	Descrição
ACCEPT	-	Aceita um pacote.
REJECT	-	Rejeita um pacote.
REJECT	--reject-with type	Rejeita um pacote e envia um pacote ICMP type para a origem.
DROP	-	Descarta silenciosamente o pacote.
LOG	-	Loga os headers do pacote.
LOG	--log-level X	Loga utilizando o nível X (veja o arquivo syslog.conf).
LOG	--log-prefix	Permite acrescentar um prefixo de 29 letras. É útil para identificar o tipo pacote/regra.

As tabelas a seguir (Tabela 2 e Tabela 3) apresentam os complementos e condições utilizadas em uma sintaxe.

Tabela 2. Complementos de comandos.

Opção (short)	Opção (long)	Parâmetro	Descrição
-s	--src	X.X.X.X/Y	Endereço de origem do pacote
-d	--dst	X.X.X.X/Y	Endereço de destino do pacote
-i	--in-interface	interface	Interface pela qual o pacote chegou
-o	--out-interface	interface	Interface pela qual o pacote vai sair
-p	--protocol	tcp, udp, icmp	Protocolo do pacote
--sport	--source-port	porta[:porta]	Porta ou intervalo de portas de origem
--dport	--destination-port	porta[:porta]	Porta ou intervalo de portas de destino
-	--icmp-type	tipo/código	Pacote icmp
-	--match	-	Habilita o modo de opções estendido

Tabela 3. Parâmetros de opções (long).

Opção	Parâmetro	Descrição
--limit	X/time	Limita a média de matchs. Time pode ser /second, /minute, /hour ou /day
--limit-burst	X	Número máximo inicial de matchs. Esse valor é "recarregado" em 1 a cada intervalo de tempo em que o limite não foi alcançado, até o valor especificado.
--mask	X	Casa com um pacote "marcado" com o valor X.
--state	NEW,ESTABLISHED, RELATED, INVALID	Casa com o estado de uma conexão.
--tcp-flags	MASK FLAGS	Examina as flags "MASK" e casa com FLAGS. Exemplo: SYN,ACK ACK = examina SYN e ACK e casa com pacotes com SYN desligado e ACK ligado.
--tos	Name	Casa com o campo Type of Service.
--ttl	Valor	Casa com o valor do campo ttl do pacote

Segue um exemplo de sintaxe para uma regra específica:

#iptables -A INPUT -p TCP -s 0.0.0.0/0 --sport 80 -j ACCEPT
#iptables -A OUTPUT -p TCP -d 0.0.0.0/0 --dport 80 -j ACCEPT

Nesta sintaxe, é permitido todo o tráfego (INPUT, OUTPUT), ou seja, de entrada e saída no protocolo TCP da rede 0.0.0.0/0 (todos os endereços válidos da rede) através da porta 80, assim sendo, permite-se o tráfego HTTP (NETFILTER.ORG).

Devido às necessidades e configurações do protocolo IPv6, este trabalho focará nas informações e configurações dos parâmetros da tabela *Filter.* Existem alguns softwares que implementam uma interface gráfica para o Iptables, porém não são tão populares e na maioria dos casos acabam limitando a funcionalidade do mesmo (NETO, 2004).

2.3 Chiron

A BruCON é uma conferência anual de segurança organizada na Bélgica que promove discussões abertas sobre questões críticas de informação, privacidade, tecnologia da informação e suas implicações culturais/técnicas na sociedade (BRUCON, 2014). E foi na BruCON de 2014 que um nome muito conhecido pelos entusiastas de segurança IPv6, Antonios Atlasis, apresentou a primeira versão de sua então mais nova ferramenta intitulada Chiron.

O Chiron nada mais é do que um framework de código aberto que foi projetado para avaliação de segurança/teste de penetração em redes IPv6. Ao invés de suportar ataques mais conhecidos contra o IPv6, como outros kits de ferramentas conhecidos (Kali Linux e Anubis, por exemplo) suportam, oferece aos seus usuários a capacidade de construir quase todos os tipos de pacotes IPv6 arbitrariamente e, portanto, lançar qualquer tipo de ataques-IPv6 relacionados, conhecidos ou outros ataques que seus usuários possam imaginar. O foco principal da ferramenta é realmente sobre os cabeçalhos de extensão IPv6, um recurso IPv6 discutido bastante na literatura de segurança, mas que até agora não havia nenhuma ferramenta para explorá-los facilmente e em toda a sua extensão. (ATLASIS, 2014)

Esta ferramenta de ataque-IPv6 é escrita em Python e é baseada em Scapy, que roda em ambientes Linux, mas sem requerer qualquer conhecimento sobre estes elementos (Scapy e Python). É composto pelos seguintes módulos: a) um scanner IPv6, b) uma ferramenta de link local IPv6 e c) um proxy IPv4 para IPv6. Todos os módulos acima são suportados por uma biblioteca comum que permite a criação de cadeias de cabeçalho IPv6 completamente arbitrárias, fragmentadas ou não (ATLASIS, 2014). Apenas o primeiro módulo foi utilizado nos testes para manipular pacotes IPv6 e criar os cenários necessários, de forma a respaldar melhor a autenticidade dos mesmos. Os outros módulos oferecem recursos que não são necessários para os testes. Foi utilizado o Chiron em sua versão 0.7, além do Scapy e do Python devidamente instalados em suas versões mais atualizadas para suportar o funcionamento da ferramenta.

2.4 Política Default e Topologia

A política *default* configurada em ambos os firewalls é *deny all*, na qual somente o tráfego discriminado tem permissão para atravessar a máquina Firewall e todo o restante é negado por padrão. A política estabelecida permite traceroute, SSH (*Secure Shell*), HTTP (*HyperText Transfer Protocol*) e filtra os diferentes tipos de ICMPv6 (*Internet Control Message Protocol*), em conformidade com as respectivas RFCs (*Request for Comments*) e orientações do NIC.br (IETF 1998a, 1998b). Também foram testados alguns filtros com os cabeçalhos de extensão do IPv6 e pacotes IPv6 fragmentados, tendo em vista que os mesmos são importantes para a segurança do IPv6.

A Figura 2 apresenta a topologia empregada para a realização dos testes. Ambas as interfaces da máquina Firewall e os hosts 1 (Debian GNU/Linux) e 2 (Windows 7) foram configurados manualmente com os endereços IPv6 indicados na figura. A própria ferramenta Chiron disponibiliza os resultados dos testes, os quais são mostrados mais abaixo.

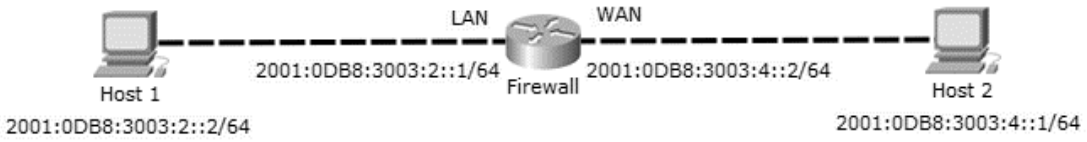

Figura 2. Topologia de rede IPv6 configurada. Fonte: Autor.

3. Resultados e Discussões

No OPNsense, a configuração da política padrão e das regras que liberam o tráfego para traceroute, SSH, HTTP e ICMPv6 é simples e facilitada pela presença da interface gráfica. É importante ressaltar que há uma opção no OPNsense que, se ativada, garante que o firewall fique acessível via HTTP na interface LAN (*Local Area Network*) sem restrição. Com essa opção desativada, o acesso HTTP ao firewall pode ser limitado com a criação de uma regra específica.

Como pode ser observado na Figura 3, o OPNsense permite o filtro dos diferentes tipos de ICMPv4:

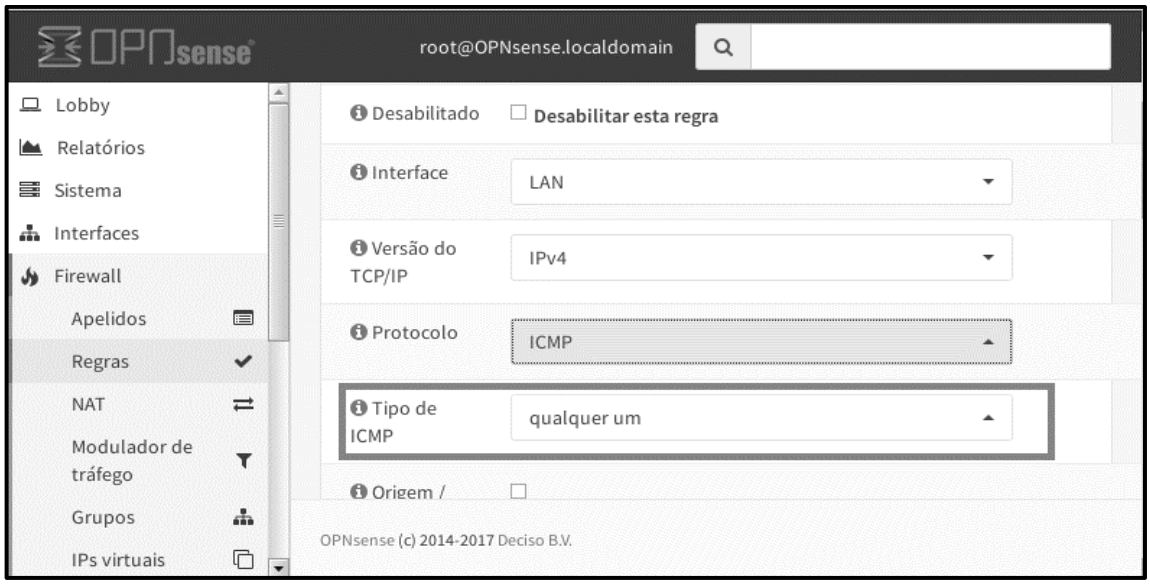

Figura 3. Configuração de regras IPv4. Fonte: Autor.

Porém, a ferramenta não permite filtrar os diferentes tipos de pacotes ICMPv6, somente admite bloquear ou liberar o tráfego ICMPv6 sem qualquer especificação, assim como mostrado na Figura 4.

Figura 4. Configuração de regras IPv6. Fonte: Autor.

Tal condição caracteriza uma importante falha de segurança no OPNsense, pois existem diversos tipos de ataques que utilizam recursos dos pacotes ICMPv6. Por outro lado, há determinados tipos de pacotes ICMPv6 que precisam ser autorizados para a correta operação do protocolo IPv6 na rede. Apesar disso, o filtro funciona corretamente e faz o que se propõe a fazer. (ARJUMAN, 2013)

O OPNsense oferece opções para filtrar alguns cabeçalhos de extensão do IPv6 como*: roteamento IPv6*, *fragmento*, e *cabeçalho de opções IPv6* (sem permitir especificar qual, podendo ser *Opções de Destino*, *Hop-by-Hop* ou ambos). Na Figura 5 são mostradas as regras configuradas na interface *Lan* do Firewall que bloqueiam o Cabeçalho de Opções do IPv6 e

IPv6 fragmentado. Há também uma regra liberando o protocolo HTTP, com o objetivo de permitir o acesso à interface gráfica do software.

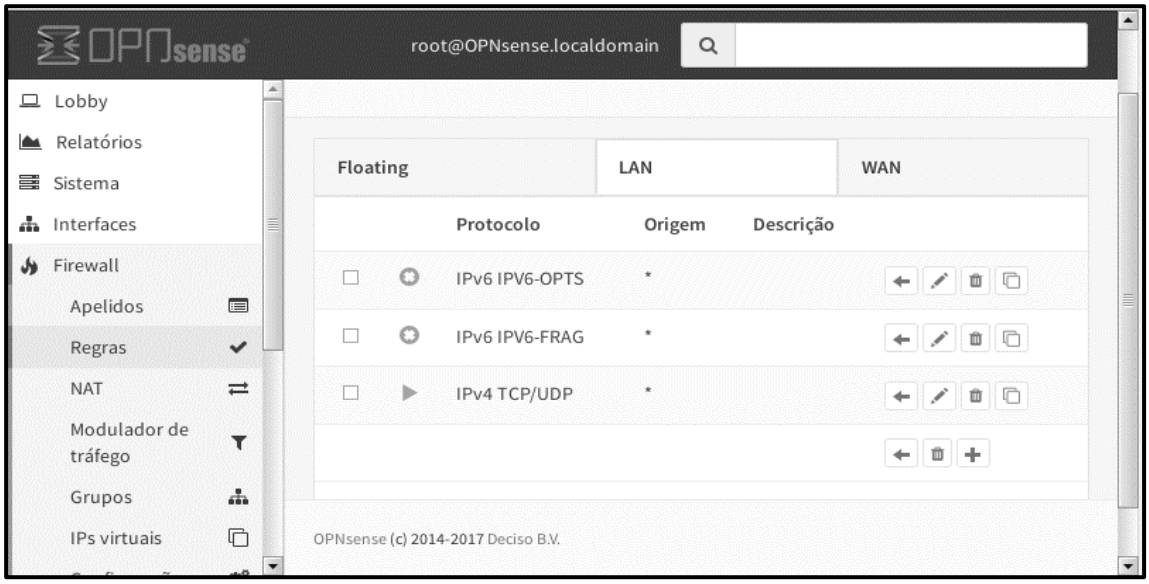

Figura 5. Regras na Interface Lan. Fonte: Autor.

O OPNsense possui uma aba para cada interface, sendo que as regras são criadas e aplicadas por interface. Porém, existe uma aba denominada Floating, onde podem ser configuradas as chamadas regras flutuantes. Estas regras podem ser aplicadas em qualquer interface, e até em mais de uma. Foi configurada uma regra nesta aba (Figura 6) que tem como objetivo liberar todo o tráfego IPv6 que passa pelo firewall.

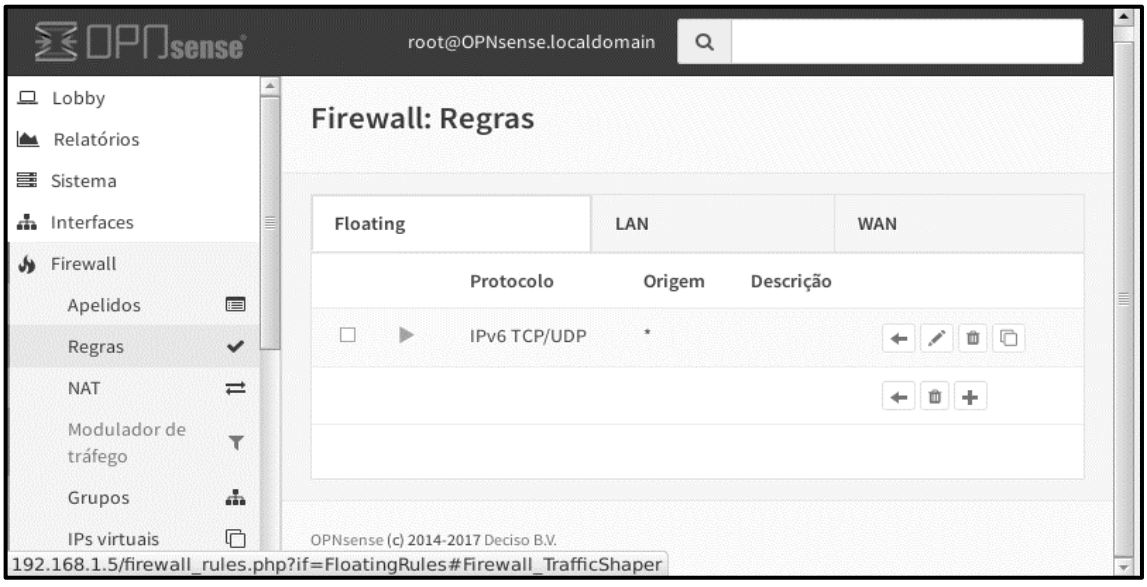

Figura 6. Regra na aba Floating. Fonte: Autor.

É importante ressaltar que deve se deixar desmarcado a opção *Quick Match*, para que as regras aplicadas individualmente em cada interface, tenham prioridade sobre as regras aplicadas na aba *Floating*.

A sintaxe do comando que utiliza o Chiron é muito simples, e o mesmo deve ser executado dentro da pasta da ferramenta. Segue abaixo uma breve explicação de cada elemento utilizado nos testes acima:

- **chiron_scanner.py**: Chama o modulo "scanner" do Chiron, que é utilizado nos testes;
- **eth0**: Especifica em qual interface o teste será executado;
- **-gw**: Parâmetro que especifica o endereço de gateway por onde passará o teste;
- **-d**: Parâmetro que especifica o host destino do teste;
- **-sS**: Parâmetro que especifica que será enviado um SYN TCP no teste;
- **-lfE**: Parâmetro que especifica que o pacote será fragmentado;
- **-luE**: Parâmetro que especifica que o pacote não será fragmentado;
- **-nf (numero)**: Parâmetro que especifica o número de fragmentos.

Apesar da configuração demonstrada mais acima, nos testes realizados e mostrados na Figura 7, o filtro de Cabeçalho de Opções não funcionou e os pacotes, mesmo marcados para bloqueio no firewall, foram recebidos pelo Host 2, o que indica mais uma falha de segurança da ferramenta (ATLASIS; REY; SCHAEFER, 2014). O Chiron não retornaria nenhum resultado, como os mostrados nas linhas 11, 12 e 22, da Figura 7, caso o firewall realmente bloqueasse os pacotes.

```
1.  root@Leonardo:~/Downloads/Chiron_0.7/Chiron/bin# ./chiron_scanner.py eth0 -gw
2.  2001:db8:3003:2::1 -d 2001:db8:3003:4::1 -sS -luE 0
3.  The MAC address of your sender is: 08:00:27:5e:8f:31
4.  The IPv6 address of your sender is: 2001:db8:3003:2::2
5.  The interface to use is eth0
6.  Starting sniffing...
7.  Sniffer filter is ip6 and dst 2001:db8:3003:2::2 and not host 2001:470:20::2
8.  The MAC address of your gateway is 08:00:27:52:9b:9c
9.  Let's start scanning
10. Press Ctrl-C to terminate before finishing
11. 08:00:27:52:9b:9c 2001:db8:3003:4::1 08:00:27:5e:8f:31 2001:db8:3003:2::2 TCP
12. http 13162 SA
13.
14.
15. Scanning Complete!
16. ==================
17.
18.
19. IPv6 address                              Protocol    Port        Flags
20.
21.
22. ['2001:db8:3003:4::1', ' TCP ', 'http', 'SA']
23. root@Leonardo:~/Downloads/Chiron_0.7/Chiron/bin#
24. root@Leonardo:~/Downloads/Chiron_0.7/Chiron/bin#
```

Figura 7. Teste com cabeçalhos de extensão sem fragmentação. Fonte: Autor.

Também foi testado a opção de filtrar pacotes IPv6 fragmentados. Porém, assim como verifica-se na Figura 8, nos testes realizados esses filtros não funcionaram e os pacotes, mesmo marcados para bloqueio no firewall, foram recebidos pelo Host 2, o que também indica uma falha de segurança dessa ferramenta (ATLASIS; REY; SCHAEFER, 2014). Mais uma vez, o Chiron não teria retornado os resultados mostrados nas linhas 11, 12 e 22, caso o firewall bloqueasse os pacotes.

```
root@Leonardo:~/Downloads/Chiron_0.7/Chiron/bin# ./chiron_scanner.py eth0 -gw
2001:db8:3003:2::1 -d 2001:db8:3003:4::1 -sS -lfE 0 -nf 2
The MAC address of your sender is: 08:00:27:5e:8f:31
The IPv6 address of your sender is: 2001:db8:3003:2::2
The interface to use is eth0
Starting sniffing...
Sniffer filter is ip6 and dst 2001:db8:3003:2::2 and not host 2001:470:20::2
The MAC address of your gateway is 08:00:27:52:9b:9c
Let's start scanning
Press Ctrl-C to terminate before finishing
08:00:27:52:9b:9c 2001:db8:3003:4::1 08:00:27:5e:8f:31 2001:db8:3003:2::2 TCP
http 46685 SA


Scanning Complete!
==================


IPv6 address                              Protocol    Port        Flags


['2001:db8:3003:4::1', ' TCP ', 'http', 'SA']
root@Leonardo:~/Downloads/Chiron_0.7/Chiron/bin#
root@Leonardo:~/Downloads/Chiron_0.7/Chiron/bin#
```

Figura 8. Teste com cabeçalhos de extensão com fragmentação. Fonte: Autor.

Já no firewall usando Iptables, a configuração da política padrão e das regras descritas anteriormente não tem o auxílio de uma interface gráfica amigável e, somada a uma sintaxe de comandos complexa, torna-se uma tarefa dispendiosa e propensa a erros. Apesar disso, a ferramenta permite a criação de regras específicas para o tratamento de pacotes sob diversos parâmetros, como é possível observar no anexo.

O Iptables oferece opções funcionais para filtrar cabeçalhos de extensão e fragmentos do IPv6, além dos diversos tipos de pacotes ICMPv6. Nos testes realizados, ao contrário do OPNsense, os pacotes não autorizados no firewall foram bloqueados e não chegaram ao destino, assim como é possível verificar nas Figuras 9 e 10. Neste caso nenhuma resposta foi recebida pois os pacotes foram devidamente bloqueados, diferente do que aconteceu nas Figuras 7 e 8.

```
root@Leonardo:~/Downloads/Chiron_0.7/Chiron/bin# ./chiron_scanner.py eth0 -gw
2001:db8:3003:2::1 -d 2001:db8:3003:4::1 -sS -lfE 0 -nf 2
The MAC address of your sender is: 08:00:27:5e:8f:31
The IPv6 address of your sender is: 2001:db8:3003:2::2
The interface to use is eth0
Starting sniffing...
Sniffer filter is ip6 and dst 2001:db8:3003:2::2 and not host 2001:470:20::2
The MAC address of your gateway is 64:1c:67:85:f4:ca
Let's start scanning
Press Ctrl-C to terminate before finishing



Scanning Complete!
==================


IPv6 address                              Protocol    Port        Flags


root@Leonardo:~/Downloads/Chiron_0.7/Chiron/bin#
```

Figura 9. Teste do iptables com fragmentação. Fonte: Autor.

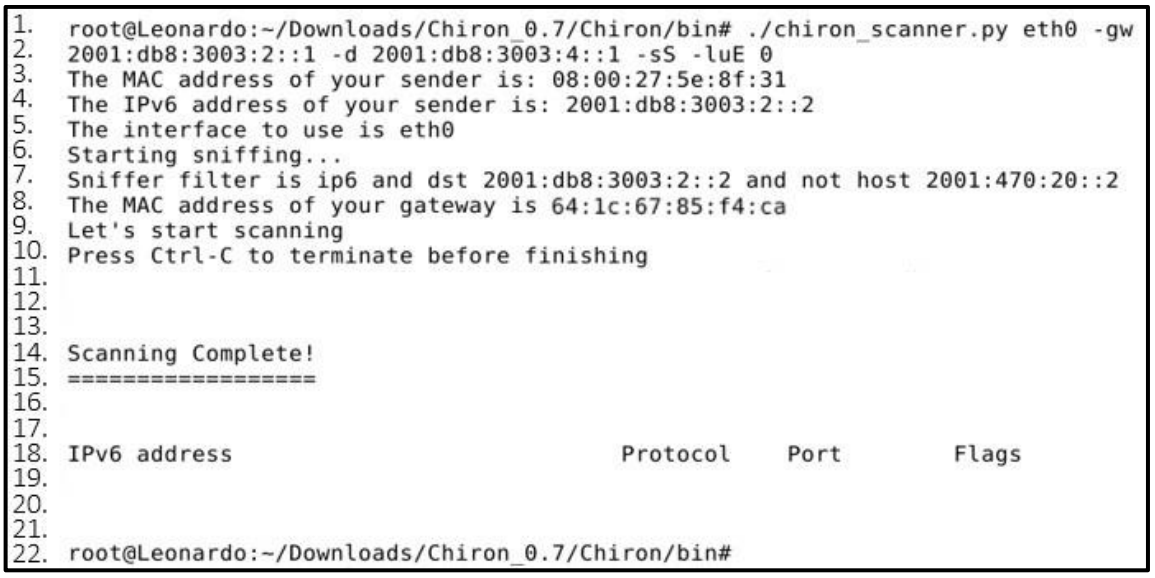

Figura 10. Teste do iptables sem fragmentação. Fonte: Autor.

4. Conclusões

Na configuração do firewall, a principal vantagem do OPNsense é a presença de uma interface gráfica amistosa e intuitiva, que facilita a criação e manipulação das regras definidas na política de segurança da rede. Entretanto, a ferramenta apresentou algumas limitações no tratamento dos recursos oferecidos pelo protocolo de endereçamento IPv6. Os filtros disponíveis no OPNsense para os fragmentos de pacotes IPv6 e o cabeçalho de opções do protocolo, mesmo habilitados, não funcionaram corretamente, assim como mostraram os testes. Além disso, outra desvantagem da ferramenta, é o fato de não aceitar especificar os variados tipos de pacotes ICMPv6 para serem filtrados. Isto por si só já é uma falha de segurança, porém o fato de o software oferecer uma função que na prática não funciona (filtros de cabeçalhos de extensão) é ainda mais grave, pois pode dar ao usuário uma falsa ideia de segurança. Essas características sinalizam falhas de segurança importantes do OPNsense no trato do protocolo IPv6.

Um ponto positivo a favor do OPNsense é que os desenvolvedores disponibilizam atualizações semanais e os mesmos estão sempre atentos aos fóruns oficiais, onde os usuários apontam erros que precisam ser corrigidos no software, dando assim a possibilidade de que o OPNsense possa melhorar constantemente.

Já o Iptables, apesar da sua configuração menos amistosa e mais onerosa por meio de linhas de comandos, mostrou-se uma solução mais flexível e segura ao admitir a implementação de filtros específicos para IPv6 e ICMPv6, sendo que todos os filtros testados funcionaram corretamente.

5. Referências

ARJUMAN, Navaneethan C. **IPv6 Security:** ICMPv6 Vulnerabilities**.** Havaí-EUA: TIP2013, (2013). Disponível em: https://www.internet2.edu/presentations/tip2013/20130116-Navaneethan-ipv6.pdf. Acesso em: 22 set. 2017.

ATLASIS, Antonios**.** Chiron – **An All-In-One IPv6 Penetration Testing Framework**. **(**2014). Disponível em: https://insinuator.net/2014/10/chiron-an-all-in-one-ipv6-penetration-testing-framework/. Acesso em: 22 set. 2017.

ATLASIS, Antonios; REY, Enno; SCHAEFER, Rafael**. Evasion of High-End IDPS Devices at the IPv6 Era**. (2014). Disponível em: https://www.blackhat.com/docs/us-

14/materials/us-14-Atlasis-Evasion-Of-HighEnd-IPS-Devices-In-The-Age-Of-IPv6-WP.pdf Acesso em: 22 set. 2017.

ATLASIS, Antonios. **Security Impacts of Abusing IPv6 Extension Headers**. (2012). Disponível em: https://media.blackhat.com/ad-12/Atlasis/bh-ad-12-security-impacts-atlasis-wp.pdf. Acesso em: 16 nov. 2017

BRITO, S. H. B. **IPv6**: O novo protocolo da internet. 1 Ed. São Paulo: Novatec Editora LTDA, (2013).

BRUCON. **Frequently Asked Questions**: What is BruCON? (2014). Disponível em: http://2014.brucon.org/index.php/Frequently_Asked_Questions.html. Acesso em: 22 set. 2017.

INTERNET ENGINEERING TASK FORCE (IETF). **RFC 2460:** Internet Protocol, Version 6 (IPv6) Specification. (s. I.): Internet Engineering Task Force, 1998a. 39 p. Disponível em: https://tools.ietf.org/html/rfc2460. Acesso em: 22 set. 2017.

INTERNET ENGINEERING TASK FORCE (IETF). **RFC 4443:** Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification (s. I.): Internet Engineering Task Force, 1998b. 39 p. Disponível em: https://tools.ietf.org/html/rfc4443. Acesso em: 22 set. 2017.

NETO, U. **Dominando Linux Firewall Iptables**. 1. Ed. Rio de Janeiro, Ciência Moderna, (2004).

NETFILTER.ORG. **Utilizando o Iptables**. Disponível em: https://www.netfilter.org/documentation/HOWTO/pt/packet-filtering-HOWTO-7.html. Acesso em: 16 nov. 2017.

NIC.BR. **Levantamento sobre a implantação do IPv6 ao redor do mundo**. IPv6.br, São Paulo, 29 jun. (2016). Disponível em: http://ipv6.br/post/levantamento-sobre-a-implantacao-do-ipv6-ao-redor-do-mundo/ Acesso em: 22 set. 2017.

OPNSENSE®. **ABOUT OPNsense®**. (2015). Disponível em: https://opnsense.org/about/about-opnsense/. Acesso em: 02 ago. 2017.

ANEXOS

```
#!/bin/sh

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

#Endereço iptables
iptables="/sbin/ip6tables"

#IPs das redes
ip="2001:0db8:3003:2::/64 2001.0db8:3003:4::/64"

#Política padrão Deny All (recusar todos os pacotes)
$iptables -F
$iptables -P INPUT DROP
$iptables -P OUTPUT DROP
$iptables -P FORWARD DROP

#Configuração statefull (firewall filtra com base nos estados de conexão)
$iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
$iptables -A OUTPUT -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT

#Permitir tráfego localhost
$iptables -A INPUT -i lo -j ACCEPT
$iptables -A OUTPUT -o lo -j ACCEPT

#Liberando conexões 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

#Liberando conexões HTTP
$iptables -A INPUT -p tcp -d $ip --dport 80 -j ACCEPT
$iptables -A OUTPUT -p tcp -s $ip --sport  80 -j ACCEPT

#Liberando uso do traceroute
```

```
$iptables -A INPUT -p udp --dport 33434:65535 -d $ip -j ACCEPT
$iptables -A OUTPUT -p udp --dport 33434:65535 -d $ip -j ACCEPT

#Liberando o tráfego de ICMPv6
$iptables -A OUTPUT -p icmpv6 -s $ip -j ACCEPT

#######################################################################

###### RFC 4890 #####

###### Tráfego ICMPv6 que NÃO DEVE ser DESCARTADO ######
# ECHO REQUESTS E RESPONSES (Type 128 e 129)
# =========================
$iptables -A INPUT -p icmpv6 --icmpv6-type echo-request -d $ip -j ACCEPT
$iptables -A FORWARD -p icmpv6 --icmpv6-type echo-request -d $ip -j DROP
$iptables -A FORWARD -p icmpv6 --icmpv6-type echo-reply -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
#
# REGRAS DE TESTE
#
# Bloqueia pacotes IPv6 fragmentados
$iptables -A INPUT -m ipv6header --header frag --soft -j DROP
#
# Bloqueia pacotes com cabeçalhos de extensão hop-by-hop
$iptables -A INPUT -m ipv6header --header hop --soft -j DROP
#
##### Tráfego ICMPv6 que NORMALMENTE NÃO DEVE ser DESCARTADO #####
# Mobilidade IPv6 ### Apenas as habilite se o nó 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 específicos ######
```

```
## Algumas mensagens não precisam de tratamento:
# - Router Renumbering (Type 138): Devem ser autenticadas com IPSec
#
#
## Algumas mensagens precisam de políticas especificas:
# - Redirect (Type 137): Podem oferecer riscos à segurança. Sua
# utilização deve ser estudada caso a caso.
#
#
## Mensagens ainda não definidas pela a IANA ou de uso experimental
# devem ser sempre descartadas.
## A não ser que exista um caso muito especifico na rede e que elas
# sejam utilizadas.
# Descartando tudo mais
$iptables -A INPUT -s ::/0 -j DROP
$iptables -A OUTPUT -d ::/0 -j DROP
```