



CYBERSECURITY

FORTIFICANDO OS HOSTS DA LAN

OSMANY DANTAS RIBEIRO DE ARRUDA



9

LISTA DE FIGURAS

Figura 9.1 – Topologia de referência.....	7
Figura 9.2 – Processo básico de filtragem de pacotes	8
Figura 9.3 – Sentido do fluxo	11
Figura 9.4 – Exemplo de tabela de rastreamento de conexões	13
Figura 9.5 – Cenário para testes	14
Figura 9.6 – Verificação e criação da REDE NAT	14
Figura 9.7 – Cenário de estudos	16
Figura 9.8 – Etapas básicas para construção do <i>firewall</i> com IPTables	16
Figura 9.9 – Verificação das regras residentes em memória.....	18
Figura 9.10 – Descarga de regras pré-existentes.....	18
Figura 9.11 – Ajuste das permissões do arquivo.....	19
Figura 9.12 – Ajuste das políticas de acesso	19
Figura 9.13 – Envio de pacotes ICMP	20
Figura 9.14 – Bloqueio do <i>loopback</i> pelas políticas de acesso do <i>firewall</i>	21
Figura 9.15 – Desbloqueio do <i>loopback</i>	21
Figura 9.16 – Liberação do tráfego ICMP.....	23
Figura 9.17 – Restauração da troca de pacotes ICMP	23
Figura 9.18 – Aprimoramento das regras	24
Figura 9.19 – Liberação das requisições HTTP.....	26
Figura 9.20 – Ativação da análise <i>stateful</i>	26
Figura 9.21 – Exemplo de regra no IPTables para liberação do SSH.....	32

LISTA DE QUADROS

Quadro 9.1 – Protocolos de serviço	6
Quadro 9.2 – Íntegra do <i>script</i> do <i>firewall</i>	28
Quadro 9.3 – Principais campos da Tabela de conexões do <i>firewall</i>	29
Quadro 9.4 – Listagem das regras residentes em memória	30
Quadro 9.5 – <i>Logs</i> parciais SSH e ICMP	31

EXEMPLO

LISTA DE TABELAS

Tabela 9.1 – Sumarização das sub redes	9
--	---

EMSE

SUMÁRIO

9 FORTIFICANDO OS HOSTS DA LAN	6
9.1 Visão geral	6
9.1 Filtragem de pacotes.....	8
9.1.1 Lógica geral da filtragem de pacotes	8
9.1.2 Considerações sobre as políticas de acesso.....	10
9.1.3 Considerações sobre os diferentes fluxos de pacotes no <i>firewall</i>	11
9.1.4 Considerações sobre os diferentes estados das conexões a analisar	12
9.2 Conhecendo o IPTables.....	13
9.2.1 Cenário para estudo.....	14
9.3 <i>Hands on</i>	15
9.3.1 Considerações gerais para configuração do <i>firewall</i> local	16
9.3.2 Verificação de descarga de regras pré-existentes.....	17
9.3.3 Configurando as políticas de acesso e liberando o <i>loopback</i> do <i>firewall</i>	19
9.3.4 Configurando regras específicas: pacotes ICMP	22
9.3.5 Configurando regras específicas: serviço HTTP.....	25
9.3.6 Configurando regras específicas: serviço SSH.....	27
9.3.7 Configurando regras específicas: logs de acessos.....	27
9.4 Logs, tabelas de conexões e outros.....	28
9.4.1 Tabela de conexões	28
9.4.2 Listagem das regras (Tabela FILTER)	29
9.4.3 Logs de acesso SSH.....	30
REFERÊNCIAS	33
GLOSSÁRIO.....	34

9 FORTIFICANDO OS HOSTS DA LAN

9.1 Visão geral

Certamente, o termo *firewall* não é novidade para ninguém de alguma forma relacionado à TI, normalmente, fazendo referência a um dispositivo destinado a filtrar e disciplinar o tráfego que entra ou sai de uma rede de acordo com as políticas de segurança da empresa.

As políticas de segurança são implementadas de duas formas diferentes e complementares entre si: a primeira é por **adesão**, promovida pela empresa por meio da conscientização e esclarecimento de seus colaboradores, dentre outros aspectos, em relação ao uso dos recursos da rede, como o uso da Internet.

Complementarmente, os equipamentos responsáveis por disciplinar e monitorar o uso desses recursos, especialmente os *firewalls*, serão configurados com as regras de acesso que **imporão** as determinações da política de segurança. Esses equipamentos analisam as características dos pacotes que chegam ou saem de sua(s) interface(s), executando as ações determinadas pelas políticas de segurança.

Originalmente, os *firewalls* analisavam os pacotes com base nos respectivos endereços de origem e de destino, e quando necessário, também com base no serviço requisitado, representado pelo protocolo da camada de transporte (L4) mais a porta utilizada, conforme exemplificado no quadro abaixo:

Protocolo L4	Porta	Serviço	Descrição
TCP	21	FTP	Troca de arquivos na <i>web</i>
TCP	22	SSH	Acesso remoto seguro
TCP	23	Telnet	Acesso remoto (inseguro)
TCP	53	DNS	Troca de zona
UDP			Resolução de nomes (lookup)
TCP	80	HTTP	Navegação <i>web</i>
UDP	123	NTP	Sincronização de relógios

Quadro 9.1 – Protocolos de serviço

Fonte: Elaborado pelo autor (2020)

Entretanto, com a evolução da Internet, novas funcionalidades tiveram de ser agregadas ao *firewall* original, a fim de possibilitar-lhe a identificação e mitigação das

novas ameaças que surgiram, passando então, a ser chamado de UTM (*Unified Threat Management*).

No cenário representado na figura abaixo, pode-se observar o uso de um UTM segregando a rede confiável (LAN) da rede não confiável (Internet). Assim sendo, as regras configuradas servirão para disciplinar o tráfego entre essas duas redes.

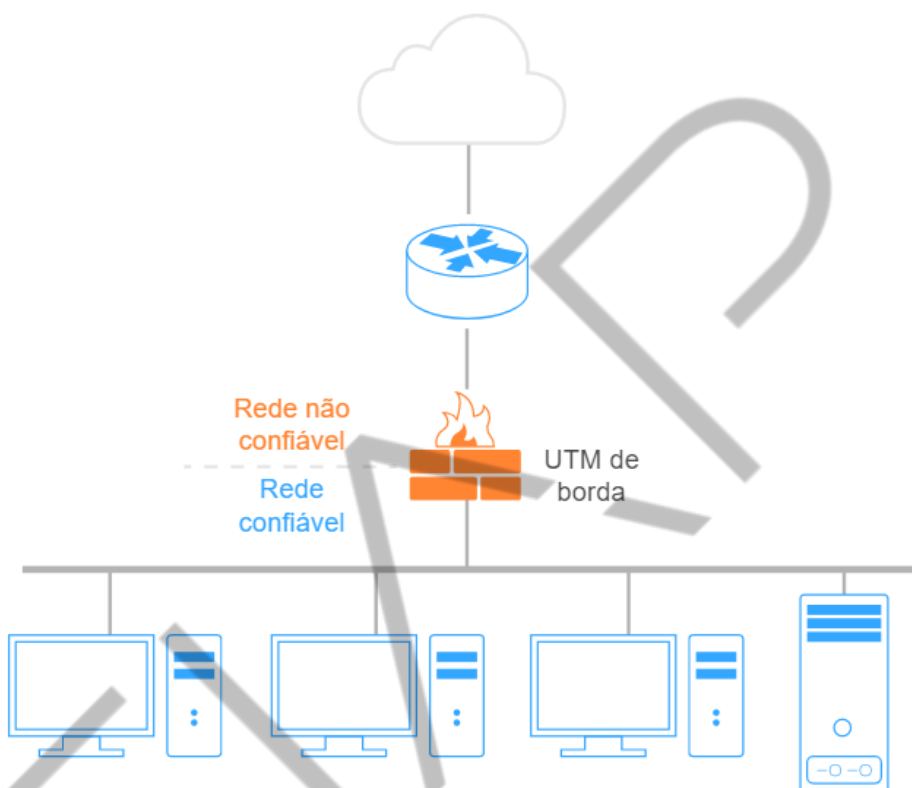


Figura 9.1 – Topologia de referência
Fonte: Elaborado pelo autor (2020)

Considere agora que alguma das estações de trabalho indicadas na figura venha a ser infectada por um *worm*, o que a levará a propagar cópias de si mesma pela rede na esperança de infectar outras estações de trabalho ou mesmo servidores.

Outra possibilidade seria um usuário mal-intencionado atacar outros hosts na rede, por exemplo, por meio de *icmp flood*. Dentre outras possibilidades, essas duas permitem observar que o UTM de borda é incapaz de proteger os *hosts* residentes na rede local em situações como essa, em que o tráfego malicioso não atravessa suas interfaces, fazendo-se necessário, portanto, que esses *hosts* possuam filtros de pacotes (*firewalls*) locais que ajudem a compor uma solução adequada à proteção do *host*.

Vale destacar que uma das opções mais eficientes recai sobre a configuração dos *firewalls* que integram as soluções de antivírus mais avançadas. Entretanto, a correta configuração da filtragem de pacotes requer considerável conhecimento técnico, a fim de evitar que falsos positivos indevidamente bloqueiem acessos legítimos e, ainda, que falsos negativos liberem um tráfego malicioso.

9.1 Filtragem de pacotes

A correta configuração da filtragem de pacotes depende de diversos fatores técnicos, os quais independem das determinações das políticas de segurança. Antes de tudo, é necessário saber a lógica utilizada por ela.

9.1.1 Lógica geral da filtragem de pacotes

A filtragem de pacotes segue uma lógica geral bastante simples, a qual deve ser claramente entendida. A figura a seguir representa o processo básico adotado pelos filtros de pacotes.

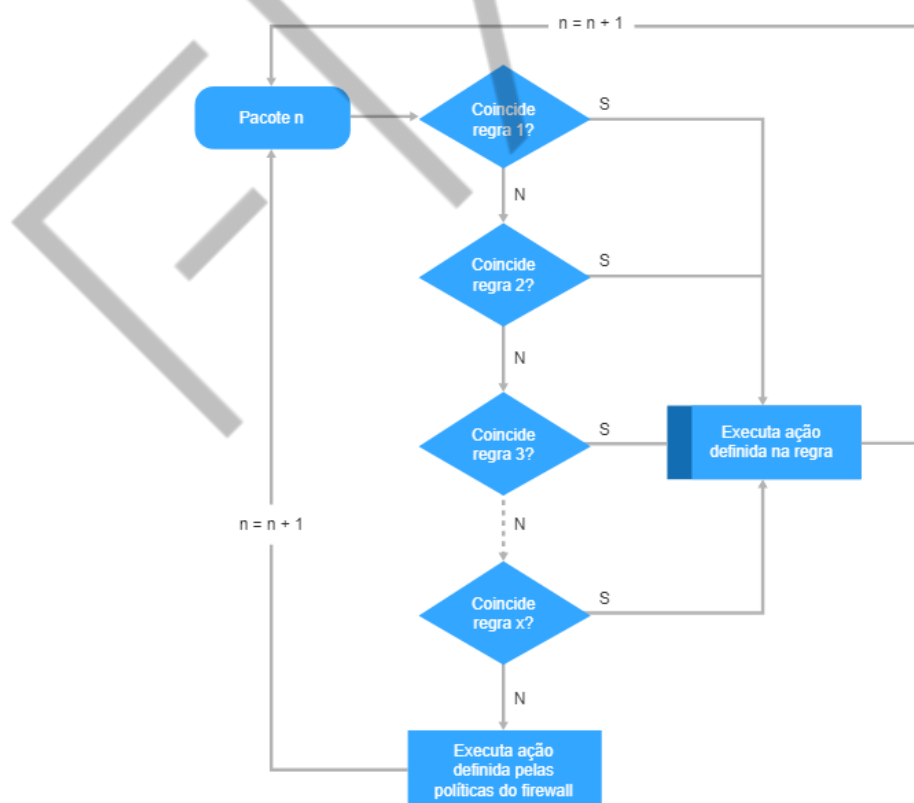


Figura 9.2 – Processo básico de filtragem de pacotes
Fonte: Elaborada pelo autor (2020)

Na figura “Processo básico de filtragem de pacotes” é possível observar que cada pacote que chega ao *firewall* é individualmente analisado. Logo, as regras de filtragem deverão ser bem planejadas, também a fim de otimizar o uso da memória (RAM), local onde as regras permanecerão residentes, em vez do disco rígido, uma vez que ele poderia causar muita latência ao processo.

Um exemplo para otimização das regras, consiste na sumarização das diversas redes de origem ou de destino, dos pacotes que deverão ser filtrados, em vez de criar uma regra para cada uma dessas redes. Admita-se que uma regra deverá permitir que apenas os *hosts* das *vlan*s 10, 20, 30 e 40 acessem um servidor.

Em vez de escrever uma regra para cada uma das sub redes dessas *vlan*s, pode-se sumarizá-las em uma *supernet*, como demonstrado na tabela a seguir, e, então, escrever uma única regra.

Tabela 9.1 – Sumarização das sub redes

<ul style="list-style-type: none"> • VLAN 10: 10.0.0.0/24 • VLAN 20: 10.0.1.0/24 • VLAN 30: 10.0.2.0/24 • VLAN 40: 10.0.3.0/24 								
	REDES						HOSTS	
	128	64	32	16	8	4	2	1
10.0.0.0	0	0	0	0	0	0	0	0
10.0.1.0	0	0	0	0	0	0	0	1
10.0.2.0	0	0	0	0	0	0	1	0
10.0.3.0	0	0	0	0	0	0	1	1
AND	0	0	0	0	0	0	0	0

Fonte: Elaborada pelo autor (2020)

Supernet: 10.0.0.0/22

No filtro, o pacote será comparado ao especificado pela primeira regra configurada. Se as características do pacote coincidirem com o especificado pela regra, a ação definida por ela será executada, caso contrário, o pacote será comparado à regra seguinte, e assim sucessivamente até que atingir a última regra configurada. Caso as características do pacote não coincidam com nenhuma das

regras específicas, serão então aplicadas as políticas de acesso configuradas no *firewall* e o processo será reiniciado com o pacote seguinte.

9.1.2 Considerações sobre as políticas de acesso

É bastante comum observar-se pessoal que não necessariamente pertence a área de segurança, mas precisa configurar um sistema de filtragem de pacotes, perguntar ao cliente: “O que eu devo bloquear”.

Recorrendo-se a dois dos principais pilares da segurança da informação (CID), sabe-se que: (I) a CONFIDENCIALIDADE da informação significa que ela deverá ser revelada ou estar acessível, unicamente aos que tiverem autorização para tal; (II) e a DISPONIBILIDADE é a propriedade da informação de estar disponível e acessível quando necessário, aos que tiverem a devida autorização para tal; conclui-se a segurança da informação é restritiva por padrão, assim sendo, a pergunta correta seria: “O que eu devo liberar”. Parte-se, portanto, do pressuposto que por padrão tudo deverá ser bloqueado, liberando-se o estritamente determinado pelas políticas de segurança, prática conhecida como *whitelisting*.

Em termos práticos, esse tipo de implementação pode ser um pouco mais complicado do que pode parecer, pois se dentre inúmeras outras possibilidades, um usuário precisar acessar algum *website* não contemplado pelas políticas de segurança, ele necessitará de permissão especial para isso, o que poderá atrapalhar seu fluxo de trabalho.

Caso a empresa venha a adotar a prática de *blacklisting*, ou seja, bloqueando estritamente os pontos específicos e liberando o acesso ao restante, situações imprevistas poderão impactar fortemente o desempenho dos sistemas de proteção, prejudicando sua eficiência e confiabilidade.

Assim sendo, de maneira geral, pode-se dizer que a opção mais adequada para configuração das políticas padrão de um *firewall* será torná-lo totalmente restritivo, ou seja, descartando todos os pacotes que não tenham correspondência com nenhuma regra específica.

9.1.3 Considerações sobre os diferentes fluxos de pacotes no *firewall*

A criação das regras para **filtragem** deve observar atentamente o sentido dos fluxos de pacotes que atravessam o *firewall*.

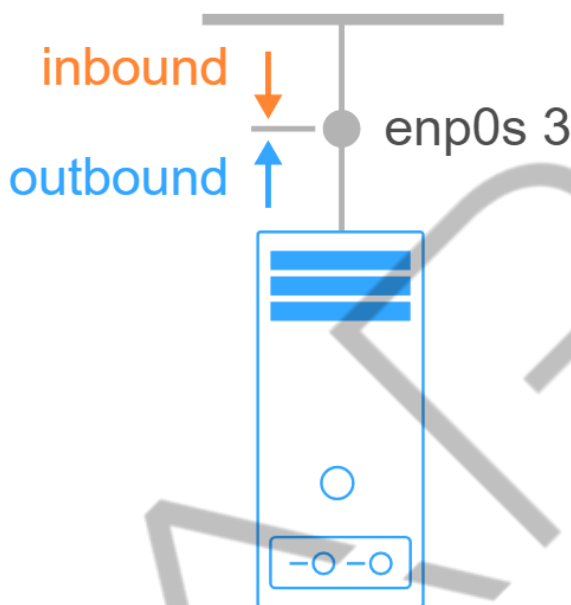


Figura 9.3 – Sentido do fluxo
Fonte: Elaborada pelo autor (2020)

Na figura acima, observa-se que a *interface* a ser monitorada pelo *firewall* deve ser tomada como referência para classificação do tráfego como entrante (*inbound*) ou saínte (*outbound*), e não a *interface* do *host* que envia a requisição ao ativo monitorado.

Vale destacar ainda que, até algum tempo atrás, muitos administradores de firewall consideravam que o tráfego que deixa o equipamento (também muitas vezes referenciado como caixa) era seguro. De maneira geral, atualmente, essa não é uma boa ideia, uma vez que a maioria dos ataques não são mais iniciados de fora para dentro da rede, mas sim, de dentro para fora.

Diante desse contexto, isso significa dizer que, além do risco de um *worm* contaminar os demais *hosts* da rede (ataque de fora para dentro), há ainda o grande risco de um usuário permitir que um *malware* entre em seu *host* via *phishing*, abrindo um *backdoor*, que não será impedido pelo *firewall* local (interno do *host*), caso a saída de pacotes deste *host* seja indiscriminadamente permitida, ou seja, um ataque de dentro para fora.

9.1.4 Considerações sobre os diferentes estados das conexões a analisar

Outro relevante aspecto a ser analisado quando da construção das regras de filtragem, diz respeito aos diferentes estados que as conexões com o *host* monitorado poderá assumir. Nos *firewalls* de primeira geração, a filtragem de pacotes priorizava a inspeção dos dados de camada 3, ou seja, endereços de origem e destino do pacote.

Embora os *firewalls* dessa geração também inspecionassem o tráfego pelas portas de serviços, ainda não monitoravam os estados das conexões, ou seja, eram *STATELESS*, mesmo que já nessa época, o principal protocolo utilizado pela camada de transporte fosse o TCP (protocolo orientado a conexão!!). Isso foi alterado na segunda geração dos *firewalls*, quando passaram a filtrar o tráfego monitorado também com base no estado das conexões (*firewalls stateful*).

Os quatro estados considerados para uma conexão são:

- **NEW:** significa que o pacote iniciou uma nova conexão ou que está de alguma outra forma associado a uma conexão na qual ainda não se tenham observado pacotes em ambas as direções (entrando e saindo das *interfaces*).
- **ESTABLISHED:** quando o pacote faz parte de uma conexão já previamente estabelecida, por exemplo, a resposta de um servidor a uma solicitação do cliente.
- **RELATED:** quando o pacote está iniciando uma nova conexão, porém, relacionada a outra conexão já existente, por exemplo, a conexão iniciada pelo FTP-DATA para transmissão dos arquivos solicitados pelo usuário (essa conexão será permitida somente se estiver relacionada a uma conexão FTP-CONTROL previamente estabelecida).
- **INVALID:** o pacote não foi identificado em nenhuma conexão existente na tabela de rastreamento de conexões (*connection tracking table*).

A tabela rastreamento de conexões é aquela na qual são registradas as informações de todas as conexões ativas no *firewall*. A figura abaixo exemplifica a tabela de rastreamento de conexões do pfSense, tradicional UTM em *software livre*.

iptables connection tracking

Legend : LAN INTERNET DMZ Wireless IPFire VPN OpenVPN

Source IP: Port Dest. IP: Port Protocol: Connection Status Expires (Secs)

All All

Update

192.168.180.214	54605	192.168.180.1	444 (SNPP)	tcp	ESTABLISHED	119:59:59
192.168.180.214	53993	217.253.242.60	444 (SNPP)	tcp	ESTABLISHED	119:59:59
89.245.253.172	58390	64.12.28.98	443 (HTTPS)	tcp	ESTABLISHED	119:59:57
192.168.181.211	46000	192.168.180.1	9091	tcp	ESTABLISHED	119:59:57
192.168.181.211	37959	192.168.180.1	9091	tcp	ESTABLISHED	119:59:57
89.245.253.172	55066	205.188.254.89	443 (HTTPS)	tcp	ESTABLISHED	119:59:57
89.245.253.172	39960	205.188.248.129	443 (HTTPS)	tcp	ESTABLISHED	119:59:57
192.168.180.214	33894	192.168.180.1	800 (MDBS_DAEMON)	tcp	ESTABLISHED	119:59:56
192.168.180.214	33891	192.168.180.1	800 (MDBS_DAEMON)	tcp	ESTABLISHED	119:59:56
192.168.180.214	48850	217.253.242.60	444 (SNPP)	tcp	ESTABLISHED	119:59:56
192.168.180.214	33892	192.168.180.1	800 (MDBS_DAEMON)	tcp	ESTABLISHED	119:59:56
192.168.180.214	50409	208.68.163.220	5222	tcp	ESTABLISHED	119:59:54
192.168.180.214	50883	192.168.180.1	139 (NETBIOS-SSN)	tcp	ESTABLISHED	119:59:53
192.168.180.214	56453	192.168.180.1	8765	tcp	ESTABLISHED	119:59:48
192.168.180.214	41298	172.28.1.162	22 (SSH)	tcp	ESTABLISHED	119:59:47
192.168.181.211	33688	192.168.181.1	445 (MICROSOFT-DS)	tcp	ESTABLISHED	119:59:47
192.168.181.211	57410	192.168.181.1	8765	tcp	ESTABLISHED	119:59:46

Figura 9.4 – Exemplo de tabela de rastreamento de conexões
Fonte: Fórum pfSense (2020)

9.2 Conhecendo o IPTables

O IPTables é um dos projetos mais populares do *netfilter.org*, consiste em uma aplicação em *userspace* destinada a administradores de sistemas, para configuração das regras para filtragem de pacotes no *kernel* Linux 2.4.x e posteriores, em linha de comando. As principais características do IPTables podem ser resumidas em:

- Listar o conjunto de regras para filtragem de pacotes.
- Gerenciar o conjunto de regras para filtragem de pacotes.
- Listar/zerar os contadores das regras de filtragem.

Aplicações avançadas do IPTables incluem ainda a construção de *clusters* de *firewalls* de alta disponibilidade, implementação de NAT para *proxies* transparentes e QoS, dentre outras.

9.2.1 Cenário para estudo

O cenário para implementação prática e estudo do *firewall* local consistirá em três máquinas virtuais: VMs Debian1 e Debian2 (clientes), e VM Metasploitable2 (servidor). O referido cenário é ilustrado abaixo.

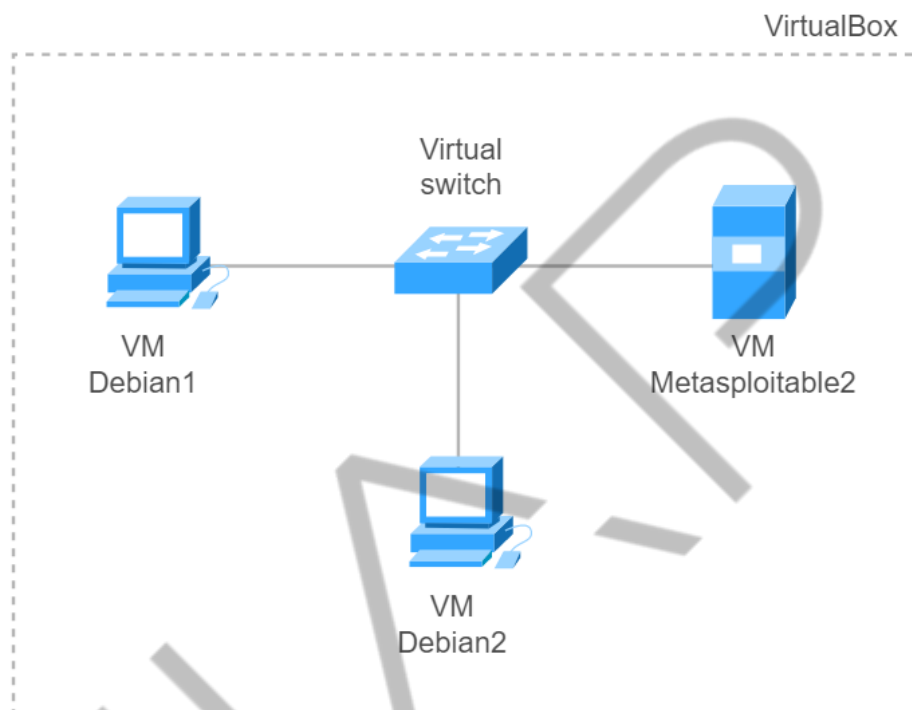


Figura 9.5 – Cenário para testes
Fonte: Elaborada pelo autor (2020)

Antes de iniciar, deve-se certificar de que o tipo de rede esteja configurado para REDE NAT. Para esse fim, no menu principal do VirtualBox, clicar em ARQUIVO > PREFERÊNCIAS > REDE (figura a seguir).

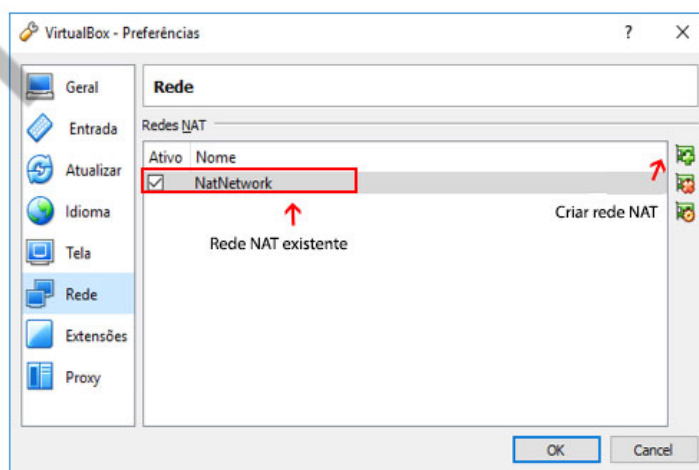


Figura 9.6 – Verificação e criação da REDE NAT
Fonte: Elaborada pelo autor (2020)

O tipo de teclado originalmente definido na VM Metasploitable também deverá ser reconfigurado de acordo com o teclado empregado pelo usuário (geralmente ABNT2). Deve ser utilizado o comando em destaque, para esse propósito, sendo que, eventualmente, algumas tentativas precisarão ser feitas antes de encontrar a configuração desejada.

```
msfadmin@metasploitable:~$ sudo dpkg-reconfigure console-setup
```

É importante destacar que a VM Metasploitable2 poderá apresentar problemas de inicialização após o ajuste no teclado.

Sempre tire *snapshot* do estado dessa VM antes de fazer **qualquer alteração**, assim, se algum problema ocorrer, ela poderá ser restaurada ao estado anterior à ocorrência.

Configurado o teclado, os endereços IP atribuídos a cada um dos *hosts* deverá ser verificado, e a conectividade entre eles testada por meio do aplicativo *ping*.

Estando tudo certo, a configuração do *firewall* local do *host* Metasploitable 2 poderá ser iniciada. No cenário, foi atribuído ao *host* Metasploitable2 o endereço IP 10.0.2.21, ao *host* Debian1 o endereço IP 10.0.2.22 e ao *host* Debian2 o endereço 10.0.2.23. Esses endereços IP poderão variar e, assim sendo, deve-se sempre utilizar os endereços efetivamente atribuídos aos *hosts* do ambiente de testes.

9.3 Hands on

Tome-se como referência para configuração do *firewall* local da VM Metasploitable2, ser ele um servidor HTTP disponível apenas para a rede local, uma espécie de INTRANET simples, sendo que inicialmente, a partir dele os usuários poderão consultar as políticas de segurança da empresa, ler avisos gerais e boletins internos dentre outras informações **devidamente classificadas pela política de segurança como públicas**. Com base nisso, adotam-se como diretrizes iniciais para configuração do *firewall* local:

- O acesso à INTRANET deverá ser feito via HTTP.

- A manutenção do servidor deverá ser feita remotamente, por meio de protocolo seguro de acesso seguro (SSH), sendo aceitas conexões originadas, exclusivamente, a partir da estação de trabalho do administrador (Debian1).
- As estações da rede local (LAN) poderão enviar pacotes ICMP ao servidor HTTP (Metasploitable2) a fim de verificar a conectividade com ele.
- Todos os acessos e pacotes ICMP enviados ao servidor deverão ser devidamente logados.

A figura abaixo ilustra o cenário considerado, com as devidas informações complementares.

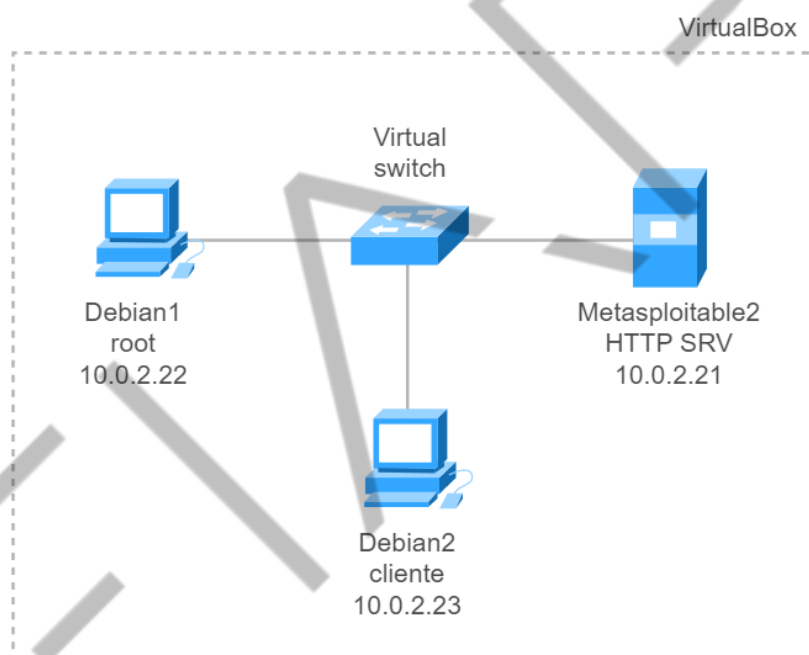


Figura 9.7 – Cenário de estudos
Fonte: Elaborada pelo autor (2020)

9.3.1 Considerações gerais para configuração do *firewall* local

A construção do *firewall* seguirá as etapas ilustradas a seguir:



Figura 9.8 – Etapas básicas para construção do *firewall* com IPTables
Fonte: Elaborada pelo autor (2020)

É importante que regras eventualmente preexistentes sejam descarregadas antes que novas sejam configuradas, a fim de assegurar que as regras residentes em memória sejam fiéis ao desejado por quem configura o equipamento. Disso vêm dois importantes pontos:

- Como todas as regras ficam residentes em memória, a fim de garantir a performance do *firewall*, é importante que elas sejam concebidas de forma eficiente. Se os *hosts* de quatro diferentes sub-redes da LAN (VLANs) devem acessar o servidor, em vez de se criar uma regra para cada uma dessas sub-redes, sempre que viável, deve-se sumariá-las criando uma única *supernet*, cujo tráfego possa ser monitorado por uma única regra de acesso.
- Toda vez que as regras do firewall forem modificadas (acrescidas, excluídas ou alteradas), o conjunto total de regras deverá ser recarregado para a memória.

Portanto, do exposto é possível observar também que, ao menos inicialmente, uma das melhores maneiras de configurar o IPTables de forma nativa, ou seja, sem linha de comando, é por meio de scripts. Isso feito, devem ser definidas as políticas de acesso a serem configuradas no *firewall*. Caso a política de segurança da empresa não indique nada em contrário, sugere-se que as políticas de acesso do *firewall* sejam totalmente restritivas, ou seja, todo tráfego que entrar ou sair do equipamento, por meio da(s) interface(s) monitorada(s) deverá ser descartado.

Entretanto, isso terá como consequência imediata o descarte também dos pacotes enviados ao *loopback*, o que poderá trazer problemas de conectividade - dentre outros, necessitando assim ser adequadamente tratado. Após tratados esses aspectos básicos, comuns a praticamente qualquer implementação, as regras específicas poderão ser, então, configuradas.

9.3.2 Verificação de descarga de regras pré-existent

Para se verificar as regras preexistentes no *host* Metasploitable2, basta solicitar ao IPTables que liste as regras residentes em memória, lembrando que, para isso, credenciais de usuários com privilégios administrativos deverão ser empregadas.

Como o Metasploitable não habilita a conta do *root*, o comando *sudo* será utilizado para escalar os privilégios do usuário *msfadmin*.

```
msfadmin@metasploitable:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
msfadmin@metasploitable:~$ _
```

Figura 9.9 – Verificação das regras residentes em memória.
Fonte: Elaborada pelo autor (2020)

O comando **iptables -L** lista as regras residentes em memória, associadas a cada um dos fluxos de dados (CHAINS) contemplados pela Tabela FILTER, Tabela padrão pelo IPTables para gerenciar o tráfego que entra (INPUT), sai (OUTPUT) e atravessa (FORWARD) o *firewall* pela interface monitorada. Como a VM Metasploitable2 possui apenas uma interface “física”, ela sempre será tomada como referência pelo IPTables.

Pela figura “Verificação das regras residentes em memória”, é possível observar que ainda não há qualquer filtragem aplicada ao tráfego que entra, sai ou atravessa o *firewall* pela *interface* monitorada. O *script* para configuração do *firewall* será iniciado criando-se no VI, ou qualquer outro editor de textos do Linux, o arquivo *firewall.sh* contendo os comandos ilustrados na figura abaixo.

```
1 #!/bin/bash
2 clear
3 #descarrega regras pre existentes
4 iptables -F
5 iptables -X
```

Figura 9.10 – Descarga de regras pré-existentes
Fonte: Elaborada pelo autor (2020)

A linha 1 no código define o *shell* a ser utilizado pelo Linux para execução do *script* de configuração do *firewall*, devendo-se atentar à sequência de caracteres que inicia a linha: **#!/**, denominado *shebang*, especificamente destinado a essa função. Na linha 2 a tela será apagada, tendo-se na linha 3, a descrição da função das linhas 4 e 5. Ambas as linhas descarregam regras preexistentes na memória, entretanto, na linha 4 são descarregadas somente regras gerais e, eventualmente, pré-configuradas

pela distribuição (-F), já na linha 5, regras específicas, criadas pelo usuário (-X). Finalizada a digitação no arquivo, ele deverá ser gravado, e suas permissões ajustadas a fim de que se torne executável conforme apresentado abaixo.

```
msfadmin@metasploitable:~$ ls -l firewall.sh
-rw-r--r-- 1 msfadmin msfadmin 76 2018-05-13 12:20 firewall.sh
msfadmin@metasploitable:~$ chmod u+x firewall.sh
msfadmin@metasploitable:~$ ls -l firewall.sh
-rwxr--r-- 1 msfadmin msfadmin 76 2018-05-13 12:20 firewall.sh
msfadmin@metasploitable:~$ _
```

Figura 9.11 – Ajuste das permissões do arquivo
Fonte: Elaborada pelo autor (2020)

Deverá ser atribuída permissão de execução do *script* ao usuário ao qual pertence, no caso, o usuário *msfadmin*. Basta utilizar o comando *chmod* conforme ilustrado na Figura “Ajuste das permissões do arquivo”. O *script* será executado ao digitar: *./firewall.sh*. À primeira vista, pode-se ter a impressão de que nada aconteceu, entretanto, se nenhum erro foi reportado, eventuais regras preexistentes foram descarregadas.

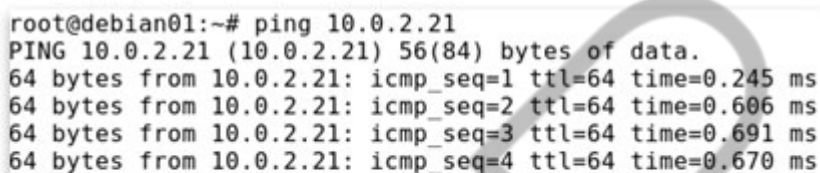
9.3.3 Configurando as políticas de acesso e liberando o *loopback* do *firewall*

Conforme apresentado anteriormente neste material, foi possível verificar que as políticas de acesso atuais do *firewall* não impõem restrições ao tráfego que adentra o *host* (*Chain INPUT*), que deixa o *host* (*Chain OUTPUT*) e nem que o atravessa (*Chain Forward*), pela interface monitorada. Assim sendo, as novas linhas ilustradas, pela figura que segue, deverão ser adicionadas ao *script*.

```
1 #!/bin/bash
2 clear
3 #descarrega regras pre existentes
4 iptables -F
5 iptables -X
6 #
7 #define as politicas de acesso do firewall
8 iptables -P INPUT DROP
9 iptables -P OUTPUT DROP
10 iptables -P FORWARD DROP
```

Figura 9.12 – Ajuste das políticas de acesso
Fonte: Elaborada pelo autor (2020)

Na linha 8, solicita-se ao *iptables* que adote como política (-P) o descarte (DROP) dos pacotes que adentrem o *firewall* (INPUT) pela interface monitorada, reforçando-se aqui, que as políticas só serão aplicadas aos pacotes cujas características não sejam previstas por nenhuma regra específica. Analogamente, nas linhas 9 e 10 são criadas as políticas aplicáveis, respectivamente, ao tráfego que deixar e que atravessa o *firewall* pela *interface* monitorada. A figura abaixo ilustra a resposta do *firewall* aos pacotes ICMP enviados pelo *host* Debian1.



```
root@debian01:~# ping 10.0.2.21
PING 10.0.2.21 (10.0.2.21) 56(84) bytes of data:
64 bytes from 10.0.2.21: icmp_seq=1 ttl=64 time=0.245 ms
64 bytes from 10.0.2.21: icmp_seq=2 ttl=64 time=0.606 ms
64 bytes from 10.0.2.21: icmp_seq=3 ttl=64 time=0.691 ms
64 bytes from 10.0.2.21: icmp_seq=4 ttl=64 time=0.670 ms
```

Figura 9.13 – Envio de pacotes ICMP
Fonte: Elaborada pelo autor (2020)

Como pode ser observado pela figura “Envio de pacotes ICMP”, ao redefinir as políticas de acesso do *firewall* como DROP, os pacotes enviados ao *host* passaram a ser descartados. Entretanto, o descarte dos pacotes pode ser feito de duas formas distintas pelo IPTables:

- DROP: nesta opção, os pacotes são descartados pelo *firewall* sem que nenhuma notificação seja enviada ao seu *host* de origem, o que faz com que ele permaneça aguardando pela resposta. Essa forma de descarte favorece a segurança, na medida em que inibe ataques por *flood*, ou seja, impede que o *host* protegido pelo *firewall* venha a ser inundado por pacotes, mitigando, assim, ataques de negação de serviço (DoS), distribuídos ou não.
- REJECT: nesta opção, o *host* de origem dos pacotes é notificado do descarte, o que faz com que o uso dessa opção seja restrito, geralmente, a testes de regras e diagnósticos, devendo ser evitado em produção.

Ao serem carregadas, as novas políticas de acesso do *firewall*, tornam também o *loopback* inacessível, o que deve ser prontamente tratado a fim de evitar os potenciais problemas decorrentes dessa situação.

```
msfadmin@metasploitable:~$ ping 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted

--- 127.0.0.1 ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 2009ms

msfadmin@metasploitable:~$ _
```

Figura 9.14 – Bloqueio do *loopback* pelas políticas de acesso do *firewall*
Fonte: Elaborada pelo autor (2020)

A figura acima ilustra a mensagem de erro retornada quando da tentativa de envio de pacotes ICMP a *interface loopback* do servidor. A fim de desbloquear o acesso a ela, duas novas regras são adicionadas ao *script*.

```
1 #!/bin/bash
2 clear
3 #descarrega regras pre existentes
4 iptables -F
5 iptables -X
6 #
7 #define as politicas de acesso do firewall
8 iptables -P INPUT DROP
9 iptables -P OUTPUT DROP
10 iptables -P FORWARD DROP
11 #libera acesso ao loopback
12 iptables -A INPUT -i lo -j ACCEPT
13 iptables -A OUTPUT -o lo -j ACCEPT
```

Figura 9.15 – Desbloqueio do *loopback*
Fonte: Elaborada pelo autor (2020)

Na linha 12, da figura acima, tem-se: **iptables -A INPUT -i lo -j ACCEPT**, a qual solicita ao IPTables que adicione (-A) uma nova regra que monitore o tráfego de entrada (INPUT) na *interface* (-i) de *loopback* (lo), liberando sua entrada (-j ACCEPT). A opção -j (*jump target*) especifica a ação a ser executada pela regra, caso as características do pacote analisado coincidam com o especificado pela regra, no caso, o tráfego deverá ser liberado (ACCEPT).

É importante observar que o uso da opção A para adicionar uma nova regra, fará com que ela seja colocada no final da lista. A posição das regras é altamente relevante em filtros de pacotes, uma vez que se colocadas em ordem inadequada, poderão produzir efeitos adversos.

Regras mais específicas deverão anteceder regras mais amplas. Por exemplo, se necessário bloquear um *host* da rede, seu endereço IP deverá ser bloqueado por uma regra, e na sequência, uma nova regra deverá liberar a rede inteira onde ele reside.

Logo, em caso de necessidade de inserção de uma nova regra em posição específica da pilha, deve-se usar a opção **-I** (*insert*) em vez de **-A** (*append*). Analogamente, a regra na linha 13 libera o tráfego de saída da *interface loopback* restaurando, assim, sua operação normal. Essa regra faz-se necessária, pois a política que supervisiona o tráfego de saída da *interface* monitorada (no caso, o *loopback*) determina seu descarte.

9.3.4 Configurando regras específicas: pacotes ICMP

Restaurado o funcionamento normal do *loopback*, é chegado o momento de se configurarem as regras específicas que constituirão o *firewall*, iniciando-se pelo monitoramento do tráfego ICMP.

Embora alguns acreditem que esse tráfego não deva ser liberado, na prática, a impossibilidade de verificação da conectividade com o servidor (via ICMP) poderá trazer inconveniências desnecessárias. Algumas aplicações poderão usar pacotes ICMP para verificar se o servidor está *online* antes de tentar conectarem-se ao serviço desejado. Em situações como essa, o bloqueio indiscriminado do tráfego ICMP poderá criar dificuldades desnecessariamente. Geralmente, essas pessoas temem que um ataque de negação de serviço venha a ser deflagrado contra um servidor com base em pacotes ICMP.

Entretanto, quando adequadamente configurados, os filtros de pacotes conseguem diferenciar entre o uso adequado desses pacotes e um ataque, não havendo, portanto, razão para essa preocupação. Assim sendo, a liberação da troca de pacotes ICMP com os servidores da rede poderá ser efetivada, não havendo, entretanto, razão para permitir a troca de pacotes ICMP entre os demais *hosts* da rede. As linhas 17 e 18 da figura abaixo exemplificam regras para liberação do tráfego ICMP.

```
1 #!/bin/bash
2 clear
3 #descarrega regras pre existentes
4 iptables -F
5 iptables -X
6 #
7 #define as politicas de acesso firewall
8 iptables -P INPUT DROP
9 iptables -P OUTPUT DROP
10 iptables -P FORWARD DROP
11 #
12 #libera acesso ao loopback
13 iptables -A INPUT -i lo -j ACCEPT
14 iptables -A OUTPUT -o lo -j ACCEPT
15 #
16 #libera trafego icmp
17 iptables -A INPUT -p icmp -j ACCEPT
18 iptables -A OUTPUT -p icmp -j ACCEPT
```

Figura 9.16 – Liberação do tráfego ICMP
Fonte: Elaborada pelo autor (2020)

Na linha 17 da figura acima, é solicitado ao IPTables que adicione uma nova regra que monitore o tráfego entrante pela *interface* monitorada, liberando a entrada (-j ACCEPT) dos pacotes do protocolo (-p) ICMP. Analogamente, a linha 18 introduz no *firewall* a regra complementar, liberando a saída dos referidos pacotes. A figura a seguir – Restauração da troca de pacotes ICMP – ilustra a volta da troca de pacotes ICMP entre a estação Debian1 e o servidor Metasploitable após o *firewall* ser recarregado com as novas regras.

```
user1@debian01:~$ ping 10.0.2.21
PING 10.0.2.21 (10.0.2.21) 56(84) bytes of data:
64 bytes from 10.0.2.21: icmp_seq=1 ttl=64 time=0.454 ms
64 bytes from 10.0.2.21: icmp_seq=2 ttl=64 time=0.291 ms
64 bytes from 10.0.2.21: icmp_seq=3 ttl=64 time=0.317 ms
64 bytes from 10.0.2.21: icmp_seq=4 ttl=64 time=0.278 ms
64 bytes from 10.0.2.21: icmp_seq=5 ttl=64 time=0.324 ms
^C
--- 10.0.2.21 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4088ms
rtt min/avg/max/mdev = 0.278/0.332/0.454/0.066 ms
user1@debian01:~$
```

Figura 9.17 – Restauração da troca de pacotes ICMP
Fonte: Elaborada pelo autor (2020)

Entretanto, deve ser observado que, embora as novas regras tenham produzido o resultado esperado, sua estruturação deve ser melhorada a fim de tornar-se devidamente granular, ou seja:

- Certamente ocorrerão casos em que o host a ser supervisionado pelo *firewall* terá mais de uma *interface*. Sendo assim, o servidor deverá aceitar pacotes ICMP provenientes de todas elas?

- O servidor deverá aceitar requisições provenientes de toda e qualquer rede que possa alcançá-lo?
- O servidor deverá aceitar qualquer tipo de pacote ICMP?

Por esses simples questionamentos, observa-se que não basta que o *firewall* “funcione”, mas que seja granular e preciso. Logo, respondendo-se ao questionamento formulado, pode-se dizer que:

- Não, o firewall deverá aceitar pacotes apenas da(s) interface(s) especificada(s), no caso: eth0.
- Não, apenas das redes especificadas.
- Não, basicamente ele deverá aceitar apenas pacotes ICMP tipo *echo reply* e *echo request*.

Logo, com base no exposto, essas regras poderiam ser otimizadas conforme ilustrado a seguir:

```
1 #!/bin/bash
2 clear
3 #descarrega regras pre existentes
4 iptables -F
5 iptables -X
6 #
7 #define as politicas de acesso firewall
8 iptables -P INPUT DROP
9 iptables -P OUTPUT DROP
10 iptables -P FORWARD DROP
11 #
12 #libera acesso ao loopback
13 iptables -A INPUT -i lo -j ACCEPT
14 iptables -A OUTPUT -o lo -j ACCEPT
15 #
16 #libera trafego icmp
17 iptables -A INPUT -i eth0 -s 10.0.2.0/24 -p icmp --icmp-type \
18     echo-request -j ACCEPT
19 iptables -A OUTPUT -o eth0 -d 10.0.2.0/24 -p icmp --icmp-type \
20     echo-reply -j ACCEPT
```

Figura 9.18 – Aprimoramento das regras

Fonte: Elaborada pelo autor (2020)

Por meio da linha 17, apresentada na figura acima, pode-se observar a especificação da *interface* (-i eth0) cujo tráfego de entrada no *firewall* deverá ser monitorado, da rede que poderá originar pacotes ICMP para o servidor (-s 10.0.2.0/24) e, finalmente, o tipo de ICMP a ser aceito (--icmp-type echo-request). Analogamente, a linha 18 descreve as condições para que os pacotes ICMP aceitos pela regra de entrada possam ser respondidos. Estritamente em relação ao IPTables tem-se:

-i: *interface* cujo tráfego entrante deverá ser monitorado, trabalha conjuntamente com a chain INPUT;

-o: *interface* cujo tráfego “sainte” deverá ser monitorado, trabalha conjuntamente com a chain OUTPUT;

-s: rede de origem (source) do tráfego monitorado;

-d: rede de destino (destination) do tráfego monitorado;

-p: protocolo (não confundir com -P, de policy).

Por fim, cabe ressaltar ser a barra invertida (\) utilizada nas regras apenas para permitir a quebra de linhas, assim, facilitando a leitura.

9.3.5 Configurando regras específicas: serviço HTTP

A liberação do serviço HTTP pode ser feita de maneira semelhante ao feito para o tráfego ICMP devendo-se, entretanto, desta vez, especificar não apenas o protocolo a ser utilizado na camada de transporte, mas também, a porta de destino da conexão.

Na linha 23 da figura “Liberação das requisições HTTP”, a opção **-p tcp** especifica o protocolo TCP para transporte do tráfego entrante à porta 80 (--dport 80), caracterizando assim, o uso do protocolo HTTP (TCP/80). Entretanto, o uso do TCP remete também ao estabelecimento de conexão. A porta de destino da conexão foi especificada como 80, entretanto, a porta de origem da conexão poderá ser qualquer uma entre 1024 e 65535, dificultando, assim, o configurar o retorno da requisição fazendo referência à porta de origem da conexão.

Uma solução adotada até pouco tempo atrás, consistia na configuração da política de OUTPUT para ACCEPT, o que permitiria que qualquer pacote saísse pelo *firewall*. Entretanto, conforme já discutido, isso poderia permitir também que códigos maliciosos tirassem proveito dessa política para abrirem *backdoors* ou mesmo se replicarem via rede.

```
1 #!/bin/bash
2 clear
3 #descarrega regras pre existentes
4 iptables -F
5 iptables -X
6 #
7 #define as politicas de acesso firewall
8 iptables -P INPUT DROP
9 iptables -P OUTPUT DROP
10 iptables -P FORWARD DROP
11 #
12 #libera acesso ao loopback
13 iptables -A INPUT -i lo -j ACCEPT
14 iptables -A OUTPUT -o lo -j ACCEPT
15 #
16 #libera trafego icmp
17 iptables -A INPUT -i eth0 -s 10.0.2.0/24 -p icmp --icmp-type \
18     echo-request -j ACCEPT
19 iptables -A OUTPUT -o eth0 -d 10.0.2.0/24 -p icmp --icmp-type \
20     echo-reply -j ACCEPT
21 #
22 #libera servico HTTP
23 iptables -A INPUT -i eth0 -s 10.0.2.0/24 -p tcp --dport 80 -j ACCEPT
```

Figura 9.19 – Liberação das requisições HTTP
Fonte: Elaborada pelo autor (2020)

A melhor opção, portanto, recai sobre o uso de *firewalls stateful*, ou seja, *firewalls* que mantêm registros das conexões permitindo, dessa forma, que pacotes que tentem sair do *firewall*, por meio de uma conexão previamente estabelecida, sejam reconhecidos como a resposta feita pela requisição que originou a conexão e, portanto, considerados como seguros, estando liberados para saírem.

```
3 #descarrega regras pre existentes
4 iptables -F
5 iptables -X
6 #
7 #define as politicas de acesso firewall
8 iptables -P INPUT DROP
9 iptables -P OUTPUT DROP
10 iptables -P FORWARD DROP
11 #
12 #libera acesso ao loopback
13 iptables -A INPUT -i lo -j ACCEPT
14 iptables -A OUTPUT -o lo -j ACCEPT
15 #
16 #habilita analise stateful
17 iptables -A OUTPUT -o eth0 -d 10.0.2.0/24 -m conntrack \
18     --ctstate ESTABLISHED -j ACCEPT
19 #
20 #libera trafego icmp
21 iptables -A INPUT -i eth0 -s 10.0.2.0/24 -p icmp --icmp-type \
22     echo-request -j ACCEPT
23 #
24 #libera servico HTTP
25 iptables -A INPUT -i eth0 -s 10.0.2.0/24 -p tcp --dport 80 -j ACCEPT
```

Figura 9.20 – Ativação da análise *stateful*
Fonte: Elaborada por autor (2020)

A ativação da análise de pacotes *stateful* é possibilitada por meio de um módulo do *kernel* denominado *conntrack*, abreviação de *connection tracking*, o qual constrói a *connection tracking table* já referenciada. A Figura “Ativação da análise *stateful*” ilustra a ativação da análise *stateful*. Na linha 17, a opção *-m (match)* especifica que pacotes que tenham registros na tabela de conexões (*conntrack*) tenham sua saída

liberada quando o fizerem através de uma conexão previamente estabelecida (--ctstate ESTABLISHED).

No caso do serviço HTTP, isso significa que a mesma conexão utilizada para solicitação da página, será usada também para sua entrega, sem a necessidade de configurar uma regra para isso. Embora o protocolo ICMP não estabeleça conexões, o módulo *conntrack* é capaz de analisar seu tráfego, razão pela qual a linha que liberava sua saída de pacotes ICMP (linha 19, Figura Liberação das requisições HTTP) também foi descartada.

9.3.6 Configurando regras específicas: serviço SSH

Analogamente ao serviço HTTP, a liberação do serviço SSH também se dará com base no protocolo da camada de transporte e porta de destino no servidor: TCP/22. Entretanto, dessa vez, apenas o *host* Debian1 poderá originar tal acesso ficando, portanto, esta regra:

```
iptables -A INPUT -i eth0 -s 10.0.2.22 -p tcp --dport 22 -j ACCEPT
```

Ao ser recarregado, o *script* permitirá, com base nessa nova linha, que acessos via SSH originados pelo *host* do administrador (*root*) possam acessar remotamente o servidor para fins administrativos.

9.3.7 Configurando regras específicas: logs de acessos

A configuração do *firewall* não estará completa se trilhas de auditoria consistentes (*logs*) não forem geradas. Os acessos (e tentativas) deverão ser logados antes mesmo de serem liberados, sendo armazenados no arquivo */var/log/messages*.

No IPTables, os *logs* são ativados por meio da *target* LOG, recomendando-se que: (a) essas sejam complementadas por identificadores que facilitem sua localização dentro do *messages*; e ainda que, (b) seja adequadamente especificado o nível de *log* a utilizar. O quadro abaixo traz a íntegra do *script* do *firewall*, incluindo a geração de logs.

```
#!/bin/bash
clear
#descarrega regras preexistentes
iptables -F
iptables -X
#
#define as politicas de acesso firewall
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP
#
#libera acesso ao loopback
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT
#
#habilita analise stateful
iptables -A OUTPUT -o eth0 -d 10.0.2.0/24 -m conntrack \
    --ctstate ESTABLISHED -j ACCEPT
#
#libera trafego icmp
iptables -A INPUT -i eth0 -s 10.0.2.0/24 -p icmp --icmp-type \
    echo-request -j LOG --log-prefix "ICMP: " --log-level info
iptables -A INPUT -i eth0 -s 10.0.2.0/24 -p icmp --icmp-type \
    echo-request -j ACCEPT
#
#libera servico HTTP
iptables -A INPUT -i eth0 -s 10.0.2.0/24 -p tcp --dport 80 -j ACCEPT
#
#libera servico SSH para o host Debian1
iptables -A INPUT -i eth0 -s 10.0.2.22 -p tcp --dport 22 -j LOG \
    --log-prefix "SSH: " --log-level info
iptables -A INPUT -i eth0 -s 10.0.2.22 -p tcp --dport 22 -j ACCEPT
```

Quadro 9.2 – Integra do *script* do *firewall*
Fonte: Elaborado pelo autor (2020)

9.4 Logs, tabelas de conexões e outros

Outros elementos relevantes e relacionados aos *firewalls*, aos quais deve-se dispensar a devida atenção são os descritos nesta seção.

9.4.1 Tabela de conexões

O quadro abaixo traz a tabela de conexões do *firewall* (*connection tracking table*) criado, a qual reside em `/proc/net/nf_conntrack`. Vale ressaltar que o acesso a esta tabela requer privilégios administrativos (root).

```

ipv4 2 udp 17 169 src=127.0.0.1 dst=127.0.0.1 sport=42011 dport=42011 packets=1 bytes=48 src=127.0.0.1 dst=127.0.0.1
sport=42011 dport=42011 packets=503 bytes=252468 [ASSURED] mark=0 secmark=0 use=1

ipv4 2 tcp 6 53 TIME_WAIT src=10.0.2.22 dst=10.0.2.21 sport=34910 dport=22 packets=24 bytes=3848 src=10.0.2.21
dst=10.0.2.22 sport=22 dport=34910 packets=24 bytes=5902 [ASSURED] mark=0 secmark=0 use=1

ipv4 2 tcp 6 74 TIME_WAIT src=10.0.2.22 dst=10.0.2.21 sport=34912 dport=22 packets=35 bytes=4356 src=10.0.2.21
dst=10.0.2.22 sport=22 dport=34912 packets=43 bytes=77482 [ASSURED] mark=0 secmark=0 use=1

```

Quadro 9.2 – Tabela de conexões (*connection tracking table*)

Fonte: Elaborado pelo autor (2020)

Tomando-se como referência a linha em destaque no quadro acima, tem-se:

Nome do protocolo	ipv4 2	
Número do protocolo	tcp 6 (6 = TCP ; 17 = UDP)	
Tempo até a entrada expirar (seg.)	74	
Estado da conexão (somente TCP)	TIME_WAIT	
Sentido	Lado que originou a conexão	Lado que recebeu a conexão
Endereço de origem da conexão	10.0.2.22	10.0.2.21
Endereço de destino da conexão	10.0.2.21	10.0.2.22
Porta de origem da conexão	34912	22
Porta de destino da conexão	22	34912
Pacotes enviados	35	43
Bytes enviados	4356	77482
[ASSURED]	se esta conexão já recebeu tráfego em ambas as direções (UDP) ou ACK em uma conexão ESTABLISHED (TCP). Caso contrário, não está presente.	

Quadro 9.3 – Principais campos da Tabela de conexões do *firewall*

Fonte: Elaborado pelo autor (2020)

9.4.2 Listagem das regras (Tabela FILTER)

O quadro “Principais campos da Tabela de conexões do *firewall*” ilustra a saída do comando ***iptables -L***, o qual lista as regras residentes na memória do *firewall* criado.

Chain INPUT (policy DROP)			
target	prot opt source	destination	
ACCEPT	all -- anywhere	anywhere	
LOG	icmp -- 10.0.2.0/24	anywhere	icmp echo-request LOG level info prefix `ICMP: '
ACCEPT	icmp -- 10.0.2.0/24	anywhere	icmp echo-request
ACCEPT	tcp -- 10.0.2.0/24	anywhere	tcp dpt:www
ACCEPT	tcp -- 10.0.2.0/24	anywhere	tcp dpt:www
LOG	tcp -- 10.0.2.22	anywhere	tcp dpt:ssh LOG level info prefix `SSH: '
Chain FORWARD (policy DROP)			
target	prot opt source	destination	
Chain OUTPUT (policy DROP)			
target	prot opt source	destination	
ACCEPT	all -- anywhere	anywhere	
ACCEPT	all -- anywhere	10.0.2.0/24	ctstate ESTABLISHED

Quadro 9.4 – Listagem das regras residentes em memória
 Fonte: Elaborado pelo autor (2020)

A coluna **target** lista a ação a ser executada por cada regra, em cada tipo de fluxo de dados (*chain*) – *INPUT*, *FORWARD* e *OUTPUT*, lembrando que eles têm como referência a interface especificada (eth0). As demais colunas são autoexplicativas.

9.4.3 Logs de acesso SSH

O quadro abaixo ilustra parcialmente os *logs* dos tráfegos ICMP e SSH criados pelo *firewall*, armazenados no arquivo */var/log/messages*.

```
May 13 20:08:27 metasploitable kernel: [13174.319421] SSH: IN=eth0 OUT=  
MAC=08:00:27:a5:22:dc:08:00:27:98:ee:12:08:00 SRC=10.0.2.22 DST=10.0.2.21 LEN=60 TOS=0x00 PREC=0x00  
TTL=64 ID=6248 DF PROTO=TCP SPT=34906 DPT=22 WINDOW=29200 RES=0x00 SYN URGP=0
```

```
May 13 20:08:27 metasploitable kernel: [13174.321065] SSH: IN=eth0 OUT=  
MAC=08:00:27:a5:22:dc:08:00:27:98:ee:12:08:00 SRC=10.0.2.22 DST=10.0.2.21 LEN=52 TOS=0x00 PREC=0x00  
TTL=64 ID=6249 DF PROTO=TCP SPT=34906 DPT=22 WINDOW=229 RES=0x00 ACK URGP=0
```

```
May 13 20:08:50 metasploitable kernel: [13197.571260] ICMP: IN=eth0 OUT=  
MAC=08:00:27:a5:22:dc:08:00:27:98:ee:12:08:00 SRC=10.0.2.22 DST=10.0.2.21 LEN=84 TOS=0x00 PREC=0x00  
TTL=64 ID=5002 DF PROTO=ICMP TYPE=8 CODE=0 ID=2011 SEQ=1
```

```
May 13 20:08:51 metasploitable kernel: [13198.578388] ICMP: IN=eth0 OUT=  
MAC=08:00:27:a5:22:dc:08:00:27:98:ee:12:08:00 SRC=10.0.2.22 DST=10.0.2.21 LEN=84 TOS=0x00 PREC=0x00  
TTL=64 ID=5081 DF PROTO=ICMP TYPE=8 CODE=0 ID=2011 SEQ=2
```

Quadro 9.5 – Logs parciais SSH e ICMP
Fonte: Elaborado pelo autor (2020)

Do destaque observa-se que às 20:08:27 horas do dia 13 de maio do ano corrente, o *host* Metasploitable (DST=10.0.2.21) recebeu uma solicitação de conexão (SYN) SSH (PROTO=TCP ; DPT=22) do *host* 10.0.2.22, originada por ele, na porta 34906 (SPT=34906). No registro seguinte, a flag ACK indica que a conexão foi executada com sucesso.

Apesar do IPTables ser uma ferramenta bastante interessante, o foco é ser mantido sobre os **conceitos abordados e não sobre a ferramenta**, pois os princípios da filtragem de pacotes podem ser aplicados a qualquer ferramenta ou *appliance*. O conhecimento adequado da pilha TCP/IP é **fundamental** para entendimento dos conceitos envolvidos

Uma interessante ferramenta para gerenciamento de *firewalls* por meio de interfaces gráficas (GUI) *drag-and-drop* é o *FirewallBuilder*. Ele consiste em uma aplicação licenciada sob o modelo GPL, capaz de configurar o IPTables, Cisco ASA/PIX e Cisco ACLs dentre outros, conforme apresentado abaixo:

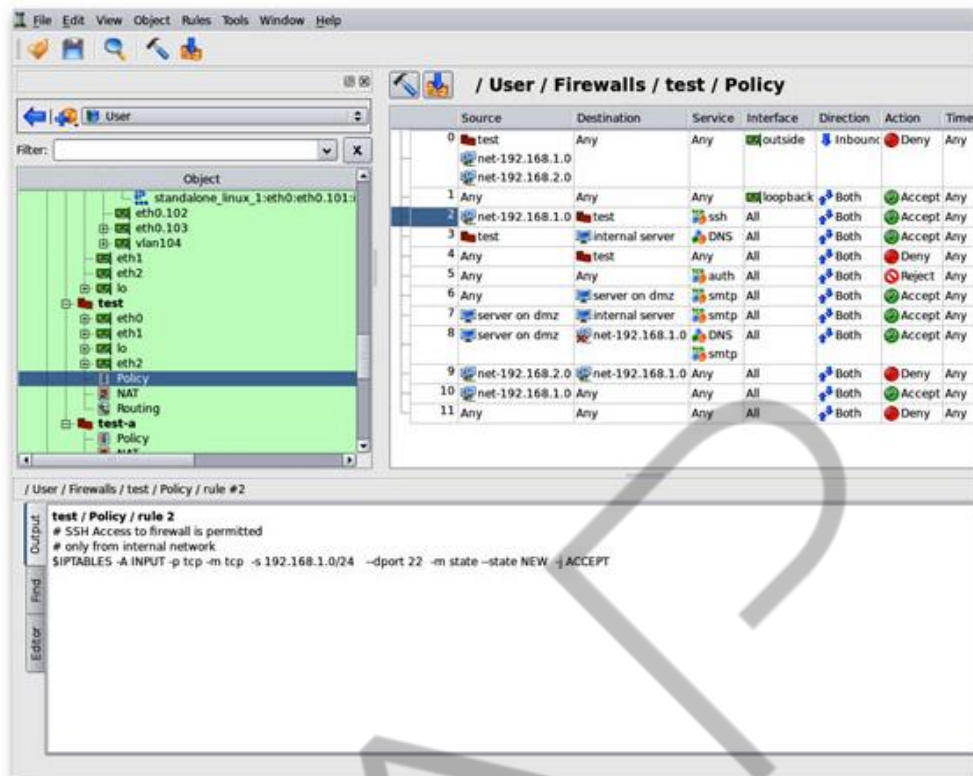


Figura 9.21 – Exemplo de regra no IPTables para liberação do SSH
 Fonte: fwbuilder.sourceforge.net (2020)

REFERÊNCIAS

AYUSO, P. N. **Netfilters's connection tracking system**. [s.d.]. Disponível em: <<http://people.netfilter.org/pablo/docs/login.pdf>>. Acesso em: 23 abr. 2020.

CISCO. **O que é um firewall**. [s.d.]. Disponível em: <https://www.cisco.com/c/pt_br/products/security/firewalls/what-is-a-firewall.html>. Acesso em: 23 abr. 2020.

FIREWALLBUILDER. [s.d.]. Disponível em: <<http://fwbuilder.sourceforge.net/index.shtml>>. Acesso em: 23 abr. 2020.

GIAC. **The packet filter: a basic network security tool**. [s.d.]. Disponível em: <<https://www.giac.org/paper/gsec/131/packet-filter-basic-network-security-tool/100197>>. Acesso em: 23 abr. 2020.

NETFILTER.ORG. **IPTables project**. [s.d.]. Disponível em: <<https://netfilter.org/projects/iptables/index.html>>. Acesso em: 23 abr. 2020.

NETFILTER.ORG. **The conntrack-tools user manual**. [s.d.]. Disponível em: <<http://conntrack-tools.netfilter.org/manual.html>>. Acesso em: 23 abr. 2020.

RASH, M. **Linux Firewalls Attack Detection and Response with IPTABLES, PSAD and FWSNORT**. San Francisco, CA.:No Starch Press, Inc. 2007.

RED HAT. **IPTables And Connection Tracking**. [s.d.]. Disponível em: <https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/4/html/Security_Guide/s1-firewall-state.html>. Acesso em: 23 abr. 2020.

GLOSSÁRIO

Worm	Código malicioso capaz de se propagar automaticamente pela rede, enviando cópias de si mesmo a outros computadores. Diferentemente dos vírus, os <i>worms</i> não embutem cópias de si mesmos em outros programas ou arquivos, nem necessitam ser explicitamente executados para se propagarem.
Loopback	<i>Interface</i> de rede virtual que permite que um cliente e um servidor no mesmo <i>host</i> se comuniquem entre si usando a pilha de protocolos TCP/IP (127.0.0.0/8). O <i>loopback</i> pode ser usado para testes de <i>software</i> ou de conectividade, permitindo verificar a funcionalidade de parte da pilha TCP/IP em uma máquina. Os pacotes enviados a uma <i>interface</i> de <i>loopback</i> nunca chegam à <i>interface</i> de rede física do <i>host</i> , o que permite inclusive testar <i>softwares</i> mesmo que o <i>host</i> não possua <i>interfaces</i> físicas.
Appliance	Dispositivo de hardware independente e dedicado, com software integrado (firmware), desenvolvido com finalidades específicas (no contexto abordado, refere-se à caixas de fabricantes como Fortinet e Barracuda Networks, dentre outros).