UNIVERSIDADE
FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

CURSO DE GRADUAÇÃO EM **ENGENHARIA DE COMPUTAÇÃO**
NONO SEMESTRE

QXD0069 - SEGURANÇA

Relatório sobre a Configuração de uma Rede Corporativa Segura no *VirtualBox*

QUIXADÁ - CE
2023

475242 — ANDERSON **SILVA SOUZA**

471047 — PEDRO HENRIQUE **MAGALHÃES BOTELHO**

473360 — SAMUEL HENRIQUE **GUIMARÃES ALENCAR**

Relatório sobre a Configuração de uma Rede Corporativa Segura no *VirtualBox*

Orientador: Prof. M.Sc. Marcos Dantas Ortiz

Relatório do trabalho final escrito para a disciplina de Segurança, no curso de graduação em Engenharia de Computação, pela Universidade Federal do Ceará (UFC), campus em Quixadá.

QUIXADÁ - CE
2023

Conteúdo

Lista de Figuras

1 Introdução

Esse é o relatório do trabalho final da disciplina de **Segurança**, ministrada pelo professor Marcos Dantas Ortiz. Nesse relatório é discutido sobre como os conhecimentos de segurança de redes (*firewall* e *proxy*) foram utilizados para construir uma rede segura para um ambiente corporativo.

Para o trabalho final, assim como nas práticas realizadas em laboratório, foi utilizado o *software* ***VirtualBox***, da **Oracle**[5], de forma a viabilizar a configuração de diversas máquinas em um mesmo ambiente controlado. As máquinas virtual executaram o sistema operacional **Ubuntu Server** [4].

O objetivo central deste trabalho é aplicar os conhecimentos adquiridos sobre *firewall* e *proxy* em um cenário de uma rede real de uma corporação, de forma a proteger a rede interna de acessos externos, e, ao mesmo tempo, permitir o acesso ao servidor *web* da organização.

Neste trabalho a equipe montou um ambiente de rede corporativa segura pelo *software* VirtualBox, onde cada máquina virtual (VM) representa um posto de trabalho da organização. O cenário da rede está documentado em Cenário da Rede da Organização.

2 Cenário da Rede da Organização

Conforme falado na Introdução, neste projeto será construído uma estrutura de uma rede segura de uma corporação. Na Figura 1 é mostrado o cenário da rede da organização, onde o **Netfilter** (Firewall) controla os acessos internos e externos à rede da corporação, e o **Squid** (Proxy) gerencia o tráfego web da rede interna da corporação.

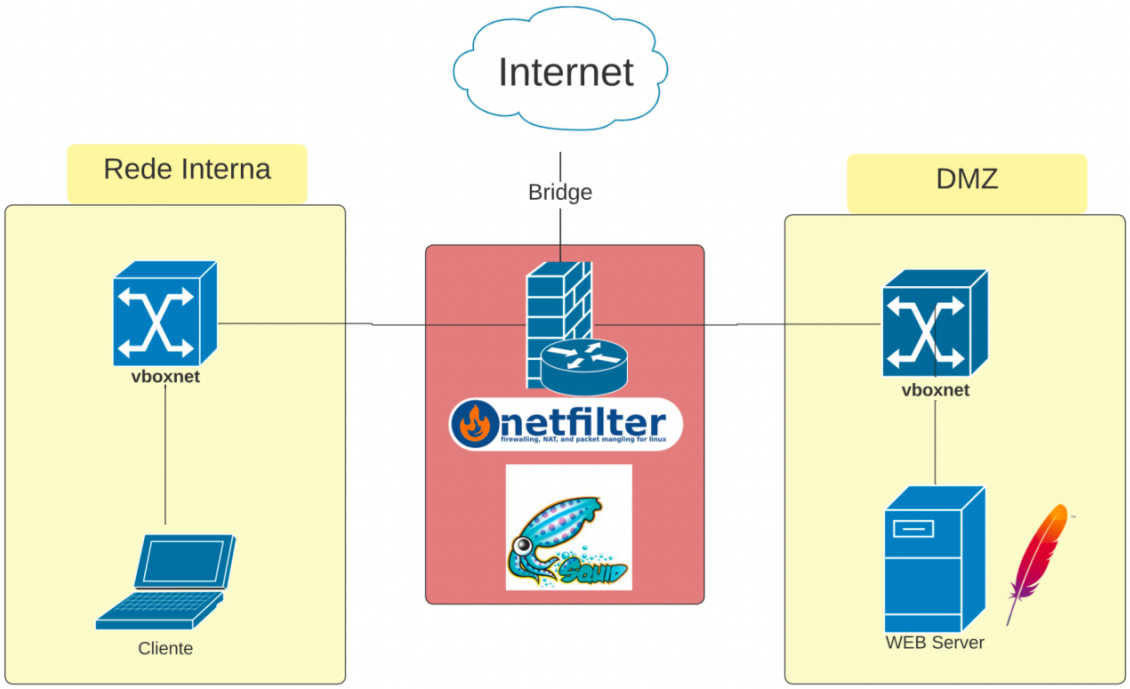

Figura 1: Cenário proposto para a rede segura da organização.

Assim como mostra a imagem é possível separar a rede da organização em três partes: **Rede Interna**, ***Firewall*** e **Zona Desmilitarizada (DMZ)**, as quais serão debatidas nesta seção.

2.1 Firewall

De acordo com [1], "*um firewall é um dispositivo de segurança que monitora o tráfego de rede de entrada e saída e decide permitir ou bloquear tráfegos específicos de acordo com um conjunto definido de regras de segurança*".

O *firewall* é implementado por meio do **Netfilter**, um subsistema do *kernel* do Linux que permite a implementação de operações de rede. Para a configuração do Netfilter é utilizado o software em linha de comando **iptables**, que provê um sistema de tabela para definir as regras do firewall para filtrar os pacotes.

Olhando por uma perspectiva estrutural, o *firewall* consiste no roteador de borda da organização, que reliza os seguintes serviços:

- Roteamento entre redes.
- Filtro de Pacotes *Stateful* com base em regras pré-estabelecidas.
- NAT (*Network Address Translation*) para mascarar os endereços IP de origem do pacote que passam pelo *firewall* (originados da rede interna da corporação).
- Proxy HTTP/HTTPS (**Squid**) restringe acesso à sites indevidos.

A *internet* é a rede mundial, pública. A organização possui duas redes diferentes que estão conectadas à internet via o *firewall*: a rede interna e a DMZ, as quais serão debatidas nas próximas seções.

2.2 Rede Interna

A rede interna da organização consiste em uma rede LAN cliente, *local area network*, uma sub-rede onde estão conectados os computadores dos funcionários da empresa. De acordo com [2], "*uma LAN, ou rede local, é um grupo de dispositivos de computação conectados em uma área localizada que geralmente compartilha uma conexão centralizada com a internet*".

Esses computadores se comunicam com a *internet*, e com o servidor *web* da organização, por meio do *firewall*, para realizar requisições, porém nunca recebendo requisições.

2.3 Zona Desmilitarizada (DMZ)

A zona desmilitarizada é outra sub-rede da organização onde fica o servidor *web* , que hospeda os recursos da organização que podem ser acessados pela internet, no caso, o site da organização.

De acordo com [3], "*uma DMZ é uma sub-rede física ou lógica que separa uma rede local interna de outras redes não confiáveis (geralmente a internet pública). Servidores, recursos e serviços externos estão localizados na DMZ. Portanto, eles são acessíveis pela Internet, mas o restante da LAN interna permanece inacessível. Isso fornece uma camada adicional de segurança à LAN, pois restringe a capacidade do hacker de acessar diretamente servidores e dados internos pela internet*".

Dessa forma o servidor *web* da organização ficará na DMZ, sendo acessível pela *internet* e pela rede local, podendo receber requisições mas nunca iniciá-las.

2.4 Configuração da Rede Segura

Tendo em mente o cenário descrito na Figura 1 e os assuntos discutidos nas seções anteriores é possível descrever a configuração das diferentes partes da rede. Já que serão necessárias

duas sub-redes, são necessárias duas interfaces de redes adicionais (além daquela conectada à *internet*). Estas são:

- **enp0s3**: A interface de ponte, denominada "bridge", atua como uma conexão direta entre a rede da máquina *host* e a rede da máquina virtual. Em termos simples, a máquina virtual faz parte da mesma rede da máquina do *host*. Dessa forma, o endereço IP do hospedeiro pode variar, pois um novo endereço IP será atribuído com base na rede à qual ele estiver conectado. Portando, não é possível determinar o IP da interface enp0s3 pois está conectada na rede do host.
- **enp0s8**: Interface *host-only* para comunicação entre a rede interna e o *firewall*, com o IP fixo 192.168.56.101.
- **enp0s9**: Interface *host-only* para comunicação entre o servidor *web* e o *firewall*, com o IP fixo 192.168.57.4.

Uma interface de rede *host-only*, como o nome sugere, apenas permite a comunicação com o *firewall*, no caso, entre uma máquina cliente e o *firewall*, ou entre o servidor *web* e o *firewall*, de forma que o *firewall* gerencie as requisições.

Conforme mencionado anteriormente na seção Firewall, o Network Address Translation (NAT) tem a função de mascarar os endereços IP das máquinas pertencentes à corporação. Isso significa que, para estabelecer comunicação com a internet, todas as estações de trabalho terão o mesmo endereço IP, ou seja, o mesmo endereço IP do firewall. Para realizar esse mascaramento do endereço IP da máquina que está tentando acessar a internet, uma regra será adicionada pelo iptables na tabela NAT do firewall após o processo de roteamento.

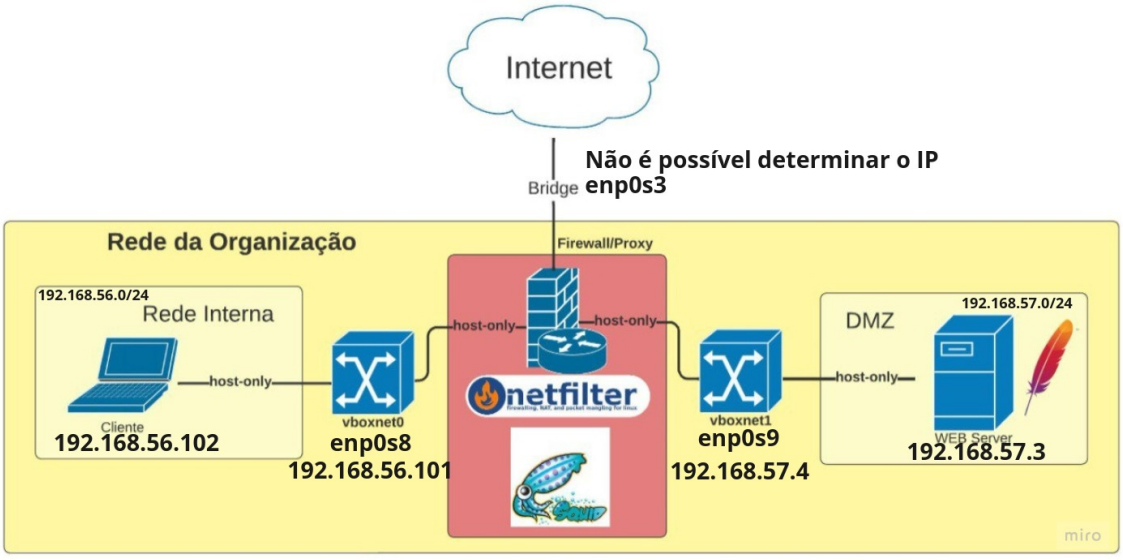

Figura 2: Configuração da rede segura da organização.

3 Regras do Firewall

Antes de configurar as regras para o *firewall* algumas variáveis são definidas:

- **IP_INTERNET="$(ip addr show enp0s3 | grep 'inet ' | cut -f2 | awk ' print $2' | cut -d/ -f1)"**

  Script utilizado para buscar o IP da interface *enp0s3*, deixando mais dinâmico o processo da regra de **PREROUTING**.

- **IP_WEB_SERVER="192.168.57.3:80"**

  Variável para o IP do servidor *web* que é fixo.

Abaixo serão listadas as regras utilizadas para a configuração do *firewall*.

3.1 Regras para o Firewall

- **sudo iptables -A INPUT -p tcp –dport 22 -m conntrack –ctstate NEW, ESTABLISHED -j ACCEPT**

  A regra especificada permite o tráfego de entrada de pacotes TCP com destino à porta 22, que é a porta padrão usada pelo protocolo SSH. A regra é adicionada à cadeia INPUT e usa a ação ACCEPT para permitir o tráfego. Além disso, a condição "–dport 22" especifica que a regra se aplica somente aos pacotes com destino à porta 22. A condição "–ctstate NEW, ESTABLISHED" garante que somente pacotes TCP com estado de conexão nova ou estabelecida sejam permitidos.

- **sudo iptables -A OUTPUT -p tcp –sport 22 -m conntrack –ctstate ESTABLISHED -j ACCEPT**

  A regra especificada permite o tráfego de saída de pacotes TCP com origem na porta 22, que é a porta padrão usada pelo protocolo SSH. A regra é adicionada à cadeia OUTPUT e usa a ação ACCEPT para permitir o tráfego. Além disso, a condição "–sport 22" especifica que a regra se aplica somente aos pacotes com origem na porta 22. A condição "–ctstate ESTABLISHED" garante que somente pacotes TCP com estado de conexão estabelecida sejam permitidos.

- **sudo iptables -P INPUT DROP**

  A regra definida permite que o *Firewall* altere a política padrão da cadeia INPUT para DROP, bloqueando todos os pacotes provenientes de qualquer protocolo e porta. Isso implica que o Firewall não permite o recebimento de pacotes, pois descarta todos eles.

- **sudo iptables -P OUTPUT DROP**

A regra especificada define a política padrão da cadeia OUTPUT do iptables como DROP, o que resulta no bloqueio de todos os pacotes de saída. Isso implica que o Firewall não permite o envio de pacotes, pois descarta todos eles.

- **sudo iptables -P FORWARD DROP**

  A regra especificada define a política padrão da cadeia FORWARD do iptables como DROP, o que resulta no bloqueio de todos os pacotes encaminhados entre interfaces de rede. Isso implica que o Firewall não permite o encaminhamento de pacotes, pois descarta todos eles.

- **sudo iptables -A INPUT -i lo -j ACCEPT**

  A regra especificada permite o tráfego de entrada na interface loopback (lo) do sistema. Isso significa que o Firewall permite a comunicação interna entre processos. Essa regra é adicionada à cadeia INPUT e utiliza a ação ACCEPT, o que permite que os pacotes sejam aceitos e não descartados.

- **sudo iptables -A OUTPUT -o lo -j ACCEPT**

  A regra especificada permite o tráfego de saída na interface loopback (lo) do sistema. Isso significa que o Firewall permite a comunicação interna entre processos. Essa regra é adicionada à cadeia OUTPUT e utiliza a ação ACCEPT, o que permite que os pacotes sejam aceitos e não descartados ao sair pela interface loopback.

- **sudo iptables -A INPUT -p icmp –icmp-type 8 -m conntrack –ctstate NEW, ESTABLISHED -j ACCEPT**

  A regra especificada permite o tráfego de entrada de pacotes ICMP do tipo 8 (Echo Request), que é comumente usado pelo comando "ping". A regra é adicionada à cadeia INPUT e usa a ação ACCEPT para permitir o tráfego. Além disso, a condição –icmp-type 8 especifica que a regra se aplica somente aos pacotes do tipo Echo Request. A condição –ctstate NEW,ESTABLISHED garante que somente pacotes ICMP com estado de conexão nova ou estabelecida sejam permitidos.

- **sudo iptables -A OUTPUT -p icmp –icmp-type 0 -m conntrack –ctstate ESTABLISHED -j ACCEPT**

  A regra especificada permite o tráfego de saída de pacotes ICMP do tipo 0 (Echo Reply), que é a resposta correspondente ao comando "ping". A regra é adicionada à cadeia OUTPUT e usa a ação ACCEPT para permitir o tráfego. Além disso, a condição "–icmp-type 0" especifica que a regra se aplica somente aos pacotes do tipo Echo Reply. A condição "–ctstate ESTABLISHED" garante que somente pacotes ICMP com estado de conexão estabelecida sejam permitidos

- **sudo iptables -A OUTPUT -p udp –dport 53 -m conntrack –ctstate NEW, ESTABLISHED -j ACCEPT**

  A regra especificada permite o tráfego de saída de pacotes UDP com destino à porta 53, que é comumente usada para comunicações DNS. A regra é adicionada à cadeia

OUTPUT e usa a ação ACCEPT para permitir o tráfego. Além disso, a condição "–dport 53" especifica que a regra se aplica somente aos pacotes com destino à porta 53. A condição "–ctstate NEW,ESTABLISHED" garante que somente pacotes UDP com estado de conexão nova ou estabelecida sejam permitidos.

- **sudo iptables -A INPUT -p udp –sport 53 -m conntrack –ctstate ESTABLISHED -j ACCEPT**

  A regra especificada permite o tráfego de entrada de pacotes UDP com origem na porta 53, que é comumente usada para comunicações DNS. A regra é adicionada à cadeia INPUT e usa a ação ACCEPT para permitir o tráfego. Além disso, a condição "–sport 53" especifica que a regra se aplica somente aos pacotes com origem na porta 53. A condição "–ctstate ESTABLISHED" garante que somente pacotes UDP com estado de conexão estabelecida sejam permitidos. Isso permite que o Firewall aceite os pacotes UDP de entrada relacionados a respostas DNS, permitindo a comunicação de retorno para solicitações DNS enviadas anteriormente.

3.2 Regras para a Rede Cliente

- **sudo iptables -A FORWARD -i enp0s8 -o enp0s3 -p tcp –dport 22 -m conntrack –ctstate NEW,ESTABLISHED -j ACCEPT**

  Essa regra permite que o cliente realize conexões SSH para qualquer lugar da internet via Firewall. Ela é colocada na tabela FORWARD do Firewall, pois ela apenas encaminha a requisição entre as duas interfaces enp0s8 e enp0s3.

  A requisição entra pela interface de rede enp0s8 (rede cliente) e sai pela interface enp0s3 (rede externa) com destino a porta 22, utilizando o protocolo TCP.

  Além disso, é utilizado o módulo *conntrack* para rastrear o estados das conexões, sendo possível apenas para conexões novas ou estabelecidas.

- **sudo iptables -A FORWARD -i enp0s8 -o enp0s3 -p udp –dport 53 -m conntrack –ctstate NEW,ESTABLISHED -j ACCEPT**

  Essa regra permite que o cliente realize consultas DNS para qualquer lugar da internet via Firewall. Neste caso, o Firewall apenas encaminhará a requisição do cliente para a rede externa, então a regra é adicionada na *chain* FORWARD. Para fazer requisições DNS é utilizado o protocolo UDP e a porta 53.

  O cliente envia uma solicitação DNS usando o comando ***nslookup***, a solicitação chega a interface enp0s8 destinado a um servidor DNS externo, o pacote é encaminhado para a interface de rede do Firewall conectada à rede externa (enp0s3), chegando até servidor. O módulo conntrack permite que os pacotes encaminhados sejam apenas de conexões novas ou estabelecidas, vale salientar que o protocolo UDP não tem estabelecimento de conexão, mas isso é simulado pelo *iptables* e módulo *conntrack*.

- **sudo iptables -A FORWARD -i enp0s8 -o enp0s3 -p tcp -m multiport –dports 80,443 -m conntrack –ctstate NEW,ESTABLISHED -j ACCEPT**

Essa regra permite que o cliente realize requisições HTTP e HTTPS para qualquer lugar da internet via Firewall. Também utilizando a *chain* FORWARD, a regra é adicionada baseada no fluxo de pacotes entre a enp0s8 e enp0s3 com destino as portas 80 e 443 que correspondem ao HTTP e HTTPS, respectivamente. Utiliza o filtro de estados para permitir que o cliente acesse apenas conexões novas ou estabelecidas.

- **sudo iptables -A FORWARD -i enp0s8 -o enp0s3 -p icmp –icmp-type 8 -m conntrack –ctstate NEW,ESTABLISHED -j ACCEPT**

  Essa regra permite que o cliente realize requisições pings para qualquer lugar da internet via Firewall. Como o cliente está tentando chegar a rede externa, a regra precisa ser adicionada na chain FORWARD, a requisição de ping é recebida pela interface do cliente (enp0s8) e chega até a rede externa (enp0s3).

  Nesta regra o tipo de ping é especificado, apenas a requisição de ping (8) é permitida e apenas para conexões novas ou estabelecidas.

- **sudo iptables -A FORWARD -i enp0s8 -o enp0s3 -p tcp -m multiport –dports 21,25 -m conntrack –ctstate NEW,ESTABLISHED -j ACCEPT**

  Essa regra permite que o cliente realize requisições FTP e SMTP utilizando o módulo multiport para qualquer lugar da internet via Firewall. Em apenas uma regra os dois serviços são permitidos, pois tanto o FTP e SMTP utilizam o protocolo TCP, alterando apenas as portas de destinos.

  Então, através da *chain* FORWARD, é permitido o tráfego de pacotes entre as interfaces enp0s8 e enp0s3 utilizando o protocolo TCP e as portas de destino 21 e 25 para conexões novas ou estabelecidas.

- **sudo iptables -A FORWARD -m conntrack –ctstate RELATED,ESTABLISHED -j ACCEPT**

  Essa regra permite que a volta dos pacotes, de qualquer protocolo e porta, de uma conexão já estabelecidas seja permitida. Essa regra facilita e enxuga bastante as *chains* do *firewall*, pois é não é preciso adicionar uma regra para a volta de cada requisição, protocolo e porta.

  Ela também evita que pacotes falsos entrem, dado que não é de uma conexão estabelecidas antes. Então pacotes vindos de fora para dentro só irão ser recebidos se pacotes de dentro para fora já tiverem saído antes.

3.3 Regras para a Rede DMZ

- **sudo iptables -t nat -A PREROUTING -d $IP_INTERNET -p tcp -m tcp –dport 80 -j DNAT –to-destination $IP_WEB_SERVER**

  Essa regra permite que o Firewall encaminhe requisições HTTP vindas da internet para o servidor WEB antes do roteamento dos pacotes.

- **sudo iptables -A FORWARD -i enp0s3 -o enp0s9 -p tcp -m tcp -m multiport –dports 80,443 -m conntrack –ctstate NEW,ESTABLISHED -j ACCEPT**

  Essa regra permite que o Firewall encaminhe requisições HTTP e HTTPS vindas da interface de internet (enp0s3) para a interface da DMZ (enp0s9). Foram utilizados filtros de estado para permitir apenas conexões novas e estabelecidas.

- **sudo iptables -A FORWARD -i enp0s8 -o enp0s3 -p tcp -m multiport –dports 80,443 -m conntrack –ctstate NEW,ESTABLISHED -j ACCEPT**

  Essa regra permite o encaminhamento (forwarding) de pacotes TCP com destino às portas 80 e 443, que são comumente usadas para os protocolos HTTP e HTTPS, entre as interfaces de rede enp0s8 e enp0s3.

- **sudo iptables -A FORWARD -i enp0s8 -o enp0s9 -p icmp –icmp-type 8 -m conntrack –ctstate NEW,ESTABLISHED -j ACCEPT**

  Permite o encaminhamento de pacotes ICMP (Internet Control Message Protocol) do tipo 8 (requisição) da rede interna (enp0s8) para o servidor *web* (enp0s9).

3.4 Regras para o *Squid*

- **sudo iptables -t nat -A PREROUTING -i enp0s8 -p tcp -m tcp –dport 80 -j REDIRECT –to-ports 3129**

  Permite que o *firewall* redirecione requisições HTTP vindas da interface de internet para a porta 3129 do SQUID. Com essa regra é possível a realização de um proxy transparente para requisições HTTP.

- **sudo iptables -t nat -A PREROUTING -i enp0s8 -p tcp -m tcp –dport 433 -j REDIRECT –to-ports 3130**

  Permite que o *firewall* redirecione requisições HTTPS vindas da interface de internet para a porta 3130 do SQUID. Com essa regra é possível a realização de um proxy transparente para requisições HTTPs.

- **sudo iptables -A INPUT -i enp0s8 -p tcp -m tcp -m multiport –dports 3129,3130 -j ACCEPT**

  Permite o tráfego de pacotes HTTP e HTTPS entre o *firewall* e o Squid, permitindo a entrada de tráfego TCP nas portas 3129 e 3130 na rede cliente interna, em enp0s8.

- **sudo iptables -A OUTPUT -o enp0s8 -p tcp -m tcp -m multiport –sports 3129,3130 -j ACCEPT**

  Permite o tráfego de pacotes HTTP e HTTPS entre o *firewall* e o Squid, permitindo a saída de tráfego TCP nas portas 3129 e 3130 na rede cliente interna, em enp0s8.

- **sudo iptables -A OUTPUT -p tcp -m multiport –dports 80,443 -m conntrack –ctstate NEW,ESTABLISHED -j ACCEPT**

  Permite o tráfego de pacotes HTTP e HTTPS entre o **Squid** e a *internet*.

- **sudo iptables -A INPUT -m conntrack –ctstate RELATED,ESTABLISHED -j ACCEPT**

  Permite o tráfego de pacotes HTTP e HTTPS entre o **Squid** e a DMZ, mas apenas permitindo requisições à DMZ.

4 Regras do *Proxy*

Abaixo estão as regras aplicadas ao *proxy* (**Squid**):

- **http_port 3129 intercept**

  Define a porta na qual o *proxy* estará interceptando requisições HTTP. Nesse caso, o *proxy* estará configurado para ouvir na porta 3129. A opção **intercept** indica que o *proxy* interceptará as solicitações HTTP e poderá tomar medidas com base em outras regras configuradas.

- **http_port 3130 intercept**

  Define uma segunda porta na qual o *proxy* estará interceptando requisições HTTPS. Nesse caso, o *proxy* estará configurado para ouvir na porta 3130. Ter várias portas permite que o *proxy* lide com mais tráfego simultâneo.

- **acl localnet src 192.168.56.0/24**

  Cria uma lista de controle de acesso (ACL) chamada ***localnet*** que inclui um intervalo de endereços IP específico (192.168.56.0/24). Essa ACL é usada para identificar um conjunto de endereços IP que serão afetados pelas regras subsequentes.

- **acl blockblacklist url_regex “/etc/squid/blacklist"**

  Cria uma segunda ACL chamada “blockblacklist” que corresponde a URLs que atendem ao padrão especificado. Nesse caso, o padrão é definido como “/etc/squid/blacklist”, indicando que o *proxy* bloqueará o acesso a qualquer URL contida no arquivo de *blacklist* localizado em “/etc/squid/blacklist”.

- **http_access deny localnet blockblacklist**

  Nega o acesso a URLs na lista de bloqueio, definida pela ACL **blockblacklist**, para os endereços IP listados na ACL "localnet". Isso significa que os clientes com endereços IP na faixa 192.168.56.0/24 serão impedidos de acessar os URLs presentes na lista de bloqueio.

- **http_access allow localnet**

  Permite que os clientes com endereços IP na faixa 192.168.56.0/24 tenham acesso ao *proxy* para todas as outras URLs que não estão na lista de bloqueio.

- **http_access allow localhost**

  Permite que o tráfego originado do próprio servidor do *proxy* (*localhost*) tenha acesso total ao *proxy*, independentemente de outras regras.

- **http_access deny all**

  Nega o acesso a qualquer outra solicitação que não tenha sido permitida pelas regras anteriores. Isso significa que qualquer cliente fora da faixa de endereços IP especificada em ***localnet*** terá seu acesso negado.

5 *Black List*

Um *blacklist*, lista negra, em um *proxy*, é uma lista de URLs, domínios, endereços IP ou palavras-chave específicas que são consideradas indesejáveis ou bloqueadas pelo *proxy*. Essa lista é usada para restringir ou proibir o acesso a determinados conteúdos ou recursos da web.

Abaixo está a lista de nomes de domínios configurados na lista negra do *proxy*:

- bet
- facebook
- .instagram.com
- .twitter.com
- .tiktok.com
- .onlyfans.com
- .privacy.com.br

6 Conclusão

Nesse trabalho foi possível entender como se dá o funcionamento e a construção de uma rede corporativa segura, entendendo a função de cada um de seus componentes (*firewall*, *proxy*, rede interna, DMZ), das regras de tráfego (do *firewall*), das interfaces de rede, etc...

Essas atividades práticas no VirtualBox (atuando como um facilitador), acompanhadas do professor orientador, surtiram grande efeito na construção de um pensamento crítico na área da segurança da informação, trazendo *insights* e experiências para os participantes.

Dito isso, e com base no material descrito neste relatório, é visível a aplicabilidade dos conhecimentos adquiridos nesta prática.

Referências

[1] O que é um firewall. https://www.sin.ufscar.br/servicos/conectividade/firewall/o-que-e-um-firewall. Accessed: 2023-06-28.

[2] O que é uma lan (rede local)? https://www.cloudflare.com/pt-br/learning/network-layer/what-is-a-lan/. Accessed: 2023-07-01.

[3] O que é uma rede dmz? https://comoaprenderwindows.com.br/infraestrutura/o-que-e-uma-rede-dmz-qual-seu-objetivo/. Accessed: 2023-07-01.

[4] Página de download do ubuntu server. https://ubuntu.com/download/server. Accessed: 2023-06-28.

[5] Página do oracle vm virtualbox. https://www.virtualbox.org/. Accessed: 2023-06-28.