# Ferramentas de Segurança em Computação

Bruno Menegaz Acerbi, Rhuan Garcia de Assis Teixeira e Pedro Igor Gomes de Morais

Universidade Federal do Espírito Santo – UFES

Centro Tecnológico

Departamento de Informática

Vitória, ES

2024

# 1 Introdução

Este trabalho tem como objetivo simular cenários de ataque em redes para desenvolver habilidades práticas na detecção e mitigação de vulnerabilidades.

Os ataques de varredura de portas serão feitos/gerados utilizando a ferramenta *Nmap*. E faremos a coleta dos dados referentes a esses ataques por meio da ferramenta *TcpDump*, capturando os pacotes de rede e possibilitando que, ao fim, seja viável utilizarmos a ferramenta *Zeek FlowMeter* para analisar esses pacotes e gerar datasets de fluxos com dados que poderão ser úteis para diferentes aplicações futuras, como o treinamento de uma IA capaz de reconhecer eventos similares em meio a um tráfego.

Nesse documento, também faremos uma breve descrição de cada ataque em seu devido tópico.

# 2 Instalação e configuração das ferramentas

Vale ressaltar que a instalação das ferramentas demonstradas nesse capítulo foram feitas sob uma versão da distro Fedora 40 Workstation Edition.

## 2.1 Instalação das Ferramentas

O primeiro passo foi a adição dos repositórios em que o *Zeek* é disponibilizado, isso porquê os pacotes do mesmo não constam nos repositórios padrão da distribuição. Isso foi feito com o comando abaixo.

```
> dnf config-manager --add-repo https://download.opensuse.org/
   repositories/security:zeek/Fedora_40/security:zeek.repo
```

Então, com todos os repositórios necessários configurados, podemos instalar as ferramentas (*nmap*, *tcpdump* e *Zeek*) utilizando o comando abaixo.

```
> sudo dnf install nmap tcpdump zeek zeek-zkg zeek-core
```

Para correção de um impeditivo relativo à falta de algumas bibliotecas Python que o *Zeek* faz uso, foi utilizado o seguinte comando.

```
> pip3 install GitPython semantic-version
```

E assim temos todos os programas necessários para a reprodução do laboratório, faltando apenas a biblioteca/extensão *FlowMeter* para o *Zeek*, que será instalada na próxima seção

## 2.2 Instalação da extensão Zeek FlowMeter

Para isso, bastou seguir o tutorial presente no github do projeto, nele recomenda-se clonar o repositório, entrar na pasta e pedir ao *zkg*, um *manager* do *zeek*, que instale a extensão. Isso pode ser feito com a seguinte sequência de comandos

```
> gh repo clone zeek-flowmeter/zeek-flowmeter
> cd zeek-flowmeter
> zkg install .
```

Assim podemos testar se tudo foi devidamente instalado com o comando

```
> zeek flowmeter
```

Que nesse caso, como não possui entrada, deve não ter nenhum retorno

# 3 Simulando o Ataque de Varredura

## 3.1 Estrutura utilizada para simular o ataque

Para realizar essa simulação de forma mais realística, condizente com um cenário comum, resolvemos utilizar como alvo um servidor caseiro feito com um notebook Dell antigo, presente no IP 192.168.0.4 executando Fedora 39 Server com seu firewall padrão (Firewalld) que está hospedando os seguintes serviços, utilizando as seguintes portas:

- SSH - 22
- Zeus admin - 9090
- BitTorrent - 6881 (serviço), 8080 (interface)
- Portainer - 9443
- Radarr - 7878
- Sonarr - 8989
- Bazarr - 6767
- Jackett - 9117
- Jellyfin - 8096

Pela forma que o firewall e os serviços estão configurados, o esperado é que apenas os serviços de SSH, BitTorrent e Zeus admin estejam sucetíveis a um scan simples.

## 3.2 Inicializando o Monitoramento da Rede

Para realizarmos o monitoramento do tráfego de pacotes na nossa rede, utilizaremos a ferramenta *TcpDump* (TCPdump & libcap, 2024), nela é possível registrar o conteúdo capturado em um arquivo do tipo.pcap.

Sendo assim, antes de iniciarmos o ataque, devemos garantir que a ferramenta escolhida estará capturando os pacotes trafegados. Por meio da seguinte chamada de terminal realizaremos a inicialização do monitoramento:

```
> sudo tcpdump -i enp9s0 -w exemplo.pcap
```

Sendo enp9s0 a interface referente a conexão cabeada utilizada para todo o tráfego do servidor e exemplo.pcap o arquivo pcap esperado como saída da captura.

A esse ponto, vale notar que foi tomada a decisão conjunta de separar os arquivos pcap, sendo cada um relativo a um ataque. Isso devido a maior utilidade vista pelo grupo, já que pouparia um desenvolvedor treinando uma IA de rotular cada linha relativa a cada ataque e também simplificaria processos em que o desenvolvedor deseje treinar um reconhecimento em específico para disparar um alerta específico por exemplo.

Além disso, também há o benefício de maior organização, e podermos ver a partir do próprio Pcap, utilizando ferramentas como Wireshark, exatamente os pacotes referentes a cada ataque sem se preocupar com timestamp, isso é relevante porquê como estamos em um cenário mais próximo do real, há um pequeno tráfego oriundo dos serviços hospedados na máquina e que caso isso não fosse feito, poderia confundir quem está fazendo a leitura do arquivo.

## 3.3 Iniciando o Ataque com o Nmap

Com o sistema de monitoramente ativado, podemos dar início aos ataques. Decediu-se por realizar 4 ataques diferentes utilizando a ferramenta *namp* (Nmap Project, 2024), cada um utilizando uma técnica distinta, sendo elas:

### 3.3.1 Varredura de porta SYN

Também conhecida como *half-open* ou *stealth scan*, essa varredura envia um pacote *SYN* e aguarda a resposta *SYN/ACK* para determinar se a porta está aberta e, ao receber a possível resposta, o *Nmap* envia um pacote *RST* para interromper a conexão. Tornando assim difícil para o alvo detectar.

```
> nmap -sS 192.168.0.4
```

### 3.3.2 Varredura de porta TCP Completa

O scan *TCP connect* é usado quando o scan *SYN* não é uma opção, como quando o usuário não tem privilégios para criar pacotes em estado bruto ou está escaneando redes *IPv6*. Neste método, o *Nmap* realiza o *handshake* de três vias completo para cada porta escaneada. Recebe-se um *RST* se a porta estiver fechada. Ele utiliza a chamada de sistema *connect()* para estabelecer conexões com as portas-alvo, assim como navegadores da web e outras aplicações de rede. Ao invés de manipular pacotes diretamente, ele utiliza a *API* de *Sockets* do sistema operacional para verificar o estado das conexões, dessa forma o *TCP connect* é mais visível nos *logs* e pode gerar alertas para administradores. Além de ser mais lento que a opção **-sS** supracitada.

```
> nmap -sT 192.168.0.4
```

### 3.3.3 Varredura de porta UDP

O scan *UDP*, ativado com a opção -sU no *Nmap*, é usado para identificar serviços que operam sobre o protocolo *UDP*, como *DNS*, *SNMP* e *DHCP*. Esse tipo de varredura é mais lento e complexo que o *TCP*, pois portas abertas e filtradas raramente respondem, e muitas vezes os hosts limitam a taxa de mensagens *ICMP*, vale citar que firewalls e filtros de rede podem bloquear facilmente pacotes UDP, dificultando uma detecção precisa.

```
> nmap -sU 192.168.0.4
```

### 3.3.4 Varredura utilizando scripts padrão

Para simularmos um cenário mais realista, decidimos por executar alguns ataques que utilizam *scripts* simples embutidos no *Nmap* para tentar detectar mais características, detalhes, sobre os serviços que utilizam as portas detectadas

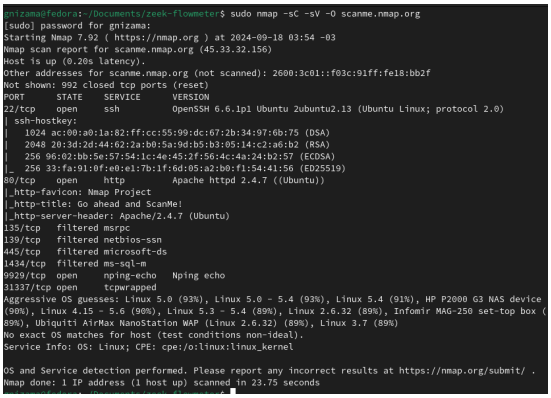
Utilizando apenas uma combinação de *scripts* comuns, padrão do *nmap*

```
> nmap -sC 192.168.0.4
```

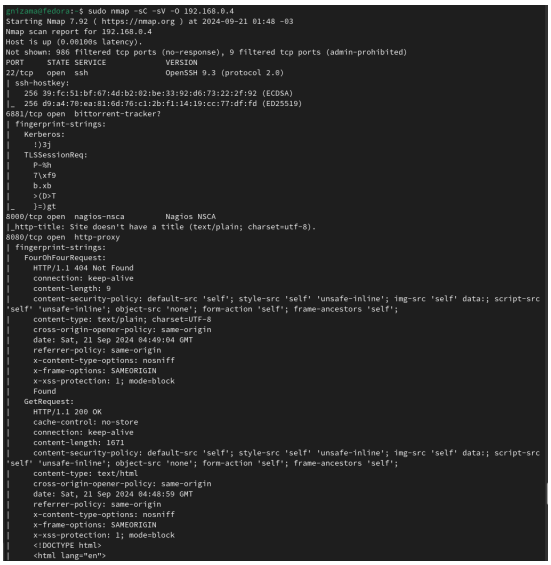
Utilizando das ferramentas do *nmap* para, além de detectar os serviços (-sC), descobrir a versão dos hosts desses serviços (-sV) e com base nisso e em outras pegadas, tentar descobrir o sistema operacional que a máquina *target* (192.168.0.4) está executando (-O)

```
> nmap -sC -sV -O 192.168.0.4
```

Veja o exemplo de output quando executado em uma máquina exemplo suscetível a esse *scan*, que possui vários serviços online em diferentes portas e o que foi obtido com o nosso scan em nossa máquina alvo. Vale notar que o print referente ao scan em nossa máquina alvo foi cortado por não caber em um único print, mas está completo no *github* do trabalho.




(a) Resultado exemplo




(b) Resultado obtido

## 3.4 Registrando os Resultados Capturados

Após a execução dos ataques, retornou-se ao processo de monitoramento (feito com o tcpdump) a fim de finalizá-lo, possibilitando obter o arquivo.pcap.

A partir daí, foi realizada a chamada da ferramenta *Zeek* (Zeek, 2024) a fim de processar os dados obtidos, analisá-los e mapeá-los para uma planilha. Para essa tarefa ser realizada, a chamada é dada da seguinte forma:

```
> zeek flowmeter -r exemplo.pcap
```

Com essa execução, um arquivo intitulado de *conn.log* será gerado contendo os dados referente ao tráfego capturado pelo *TcpDump*, agora analisado e organizado pelo *Zeek*. É possível abri-lo como um *TSV* para melhor inspeção dos dados, sendo visível, informações como ID de origem dos pacotes, protocolos, ID de destino e outras informações.

# Referências

BRUNSHWICK, U. of N. *DDoS evaluation dataset (CIC-DDoS2019)*. [S.l.], 2019.

Nmap Project. *Nmap Reference Guide*. 2024. <https://nmap.org/man/pt_PT/index.html>. Acesso em: 19 set. 2024, 21:48.

TCPdump & libcap. *TCPdump site*. 2024. <https://www.tcpdump.org/>. Acesso em: 19 set. 2024, 21:48.

Zeek. *Zeek documentation site*. 2024. <https://docs.zeek.org/en/master/>. Acesso em: 19 set. 2024, 21:48.