**Nmap Summary (ESP)**

## Parameters

### IPs to scan

* **<ip>,<net/mask>:** Indicate the ips directly
* **-iL <ips\_file>:** list\_IPs
* **-iR <number>**: Number of random Ips, you can exclude possible Ips with --exclude <Ips> or --excludefile <file>.

### Equipment discovery

By default Nmap launches a discovery phase consisting of: -PA80 -PS443 -PE -PP

* **-sL**: It is not invasive, it lists the targets making **DNS** requests to resolve names. It is useful to know if for example www.prueba.es/24 all Ips are our targets.
* **-Pn**: **No ping**. This is useful if you know that all of them are active (if not, you could lose a lot of time, but this option also produces false negatives saying that they are not active), it prevents the discovery phase.
* **-sn** : **No port scan**. After completing the reconnaissance phase, it does not scan ports. It is relatively stealthy, and allows a small network scan. With privileges it sends an ACK (-PA) to 80, a SYN(-PS) to 443 and an echo request and a Timestamp request, without privileges it always completes connections. If the target is the network, it only uses ARP(-PR). If used with another option, only the packets of the other option are dropped.
* **-PR**: **Ping ARP**. It is used by default when analyzing computers in our network, it is faster than using pings. If you do not want to use ARP packets use --send-ip.
* **-PS <ports>**: It sends SYN packets to which if it answers SYN/ACK it is open (to which it answers with RST so as not to end the connection), if it answers RST it is closed and if it does not answer it is unreachable. In case of not having privileges, a total connection is automatically used. If no ports are given, it throws it to 80.
* **-PA <ports>**: Like the previous one but with ACK, combining both of them gives better results.
* **-PU <ports>**: The objective is the opposite, they are sent to ports that are expected to be closed. Some firewalls only check TCP connections. If it is closed it is answered with port unreachable, if it is answered with another icmp or not answered it is left as destination unreachable.
* **-PE, -PP, -PM** : ICMP PINGS: echo replay, timestamp and addresmask. They are launched to find out if the target is active.
* **-PY<ports>**: Sends SCTP INIT probes to 80 by default, INIT-ACK(open) or ABORT(closed) or nothing or ICMP unreachable(inactive) can be replied.
* **-PO <protocols>**: A protocol is indicated in the headers, by default 1(ICMP), 2(IGMP) and 4(Encap IP). For ICMP, IGMP, TCP (6) and UDP (17) protocols the protocol headers are sent, for the rest only the IP header is sent. The purpose of this is that due to the malformation of the headers, Protocol unreachable or responses of the same protocol are answered to know if it is up.
* **-n**: No DNS
* **-R**: DNS always

### Port scanning techniques

* **-sS**: Does not complete the connection so it leaves no trace, very good if it can be used.(privileges) It is the one used by default.
* **-sT**: Completes the connection, so it does leave a trace, but it can be used for sure. By default without privileges.
* **-sU**: Slower, for UDP. Mostly: DNS(53), SNMP(161,162), DHCP(67 and 68), (-sU53,161,162,67,68): open(reply), closed(port unreachable), filtered (another ICMP), open/filtered (nothing). In case of open/filtered, -sV sends numerous requests to detect any of the versions that nmap supports and can detect the true state. It increases a lot the time.
* **-sY**: SCTP protocol fails to establish the connection, so there are no logs, works like -PY
* **-sN,-sX,-sF:** Null, Fin, Xmas, they can penetrate some firewalls and extract information. They are based on the fact that standard compliant machines should respond with RST all requests that do not have SYN, RST or ACK lags raised: open/filtered(nothing), closed(RST), filtered (ICMP unreachable). Unreliable on WIndows, CIsco, BSDI and OS/400. On unix yes.
* **-sM**: Maimon scan: Sends FIN and ACK flags, used for BSD, currently will return all as closed.
* **-sA, sW**: ACK and Window, is used to detect firewalls, to know if the ports are filtered or not. The -sW does distinguish between open/closed since the open ones respond with a different window value: open (RST with window other than 0), closed (RST window = 0), filtered (ICMP unreachable or nothing). Not all computers work this way, so if it is all closed, it is not working, if it is a few open, it is working fine, and if it is many open and few closed, it is working the other way around.
* **-sI:** Idle scan. For the cases in which there is an active firewall but we know that it does not filter to a certain Ip (or when we simply want anonymity) we can use the zombie scanner (it works for all the ports), to look for possible zombies we can use the scrpit ipidseq or the exploit auxiliary/scanner/ip/ipidseq. This scanner is based on the IPID number of the IP packets.
* **--badsum:** It sends the sum wrong, the computers would discard the packets, but the firewalls could answer something, it is used to detect firewalls.
* **-sZ:** "Weird" SCTP scanner, when sending probes with cookie echo fragments they should be dropped if open or responded with ABORT if closed. It can pass through firewalls that init does not pass through, the bad thing is that it does not distinguish between filtered and open.
* **-sO:** Protocol Ip scan. Sends bad and empty headers in which sometimes not even the protocol can be distinguished. If ICMP unreachable protocol arrives it is closed, if unreachable port arrives it is open, if another error arrives, filtered, if nothing arrives, open|filtered.
* **-b <server>:** FTPhost--> It is used to scan a host from another one, this is done by connecting the ftp of another machine and asking it to send files to the ports that you want to scan from another machine, according to the answers we will know if they are open or not. [<user>:<password>@]<server>[:<port>] Almost all ftps servers no longer let you do this and therefore it is of little practical use.

### **Centrar análisis**

**-p:** Sirve para dar los puertos a escanear. Para seleccionar los 65335: **-p-** o **-p all**. Nmap tiene una clasificaación interna según su popularidad. Por defecto usa los 1000 ppales. Con **-F** (fast scan) analiza los 100 ppales. Con **--top-ports <numero>** Analiza ese numero de ppales (de 1 hasta los 65335). Comprueba los puertos en orden aleatorio, para que eso no pase **-r**. También podemos seleccionar puertos: 20-30,80,443,1024- Esto ultimo significa que mire en adelante del 1024. También podemos agrupar los puertos por protocolos: U:53,T:21-25,80,139,S:9. También podemos escoger un rango dentro de los puertos populares de nmap: -p [-1024] analiza hasta el 1024 de los incluidos en nmap-services. **--port-ratio <ratio>** Analiza los puertos más comúnes que un ratio que debe estar entre 0 y 1

**-sV** Escaneado de versión, se puede regular la intensidad de 0 a 9, por defecto 7.

**--version-intensity <numero>** Regulamos la intensidad, de forma que cuanto más bajo solo lanzará las sondas más probables, pero no todas. Con esto podemos acortar considerablemente el tiempo de escaneo UDP

**-O** Deteccion de os

**--osscan-limit** Para escanear bien un host se necesita que al menos haya 1 puerto abierto y otro cerrado, si no se da esta condición y hemos puesto esto, no intenta hacer predicción de os (ahorra tiempo)

**--osscan-guess** Cuando la detección de os no es perfecta esto hace que se esfuerce más

**Scripts**

--script *<filename>*|*<category>*|*<directory>*|*<expression>*[,...]

Para usar los de por efecto vale con -sC o --script=default

Los tipos que hay son de: auth, broadcast, default, discovery, dos, exploit, external, fuzzer, intrusive, malware, safe, version, and vuln

* **Auth:** ejecuta todos sus *scripts* disponibles para autenticación
* **Default:** ejecuta los *scripts* básicos por defecto de la herramienta
* **Discovery:** recupera información del *target* o víctima
* **External:** *script* para utilizar recursos externos
* **Intrusive:** utiliza *scripts* que son considerados intrusivos para la víctima o *target*
* **Malware:** revisa si hay conexiones abiertas por códigos maliciosos o *backdoors* (puertas traseras)
* **Safe:** ejecuta *scripts* que no son intrusivos
* **Vuln:** descubre las vulnerabilidades más conocidas
* **All:** ejecuta absolutamente todos los *scripts* con extensión NSE disponibles

Para buscar scripts:

**nmap --script-help="http-\*" -> Los que empiecen por http-**

**nmap --script-help="not intrusive" -> Todos menos esos**

**nmap --script-help="default or safe" -> Los que estan en uno o en otro o en ambos**

**nmap --script-help="default and safe" --> Los que estan en ambos**

**nmap --script-help="(default or safe or intrusive) and not http-\*"**

--script-args *<n1>*=*<v1>*,*<n2>*={*<n3>*=*<v3>*},*<n4>*={*<v4>*,*<v5>*}

--script-args-file *<filename>*

--script-help *<filename>*|*<category>*|*<directory>*|*<expression>*|all[,...]

--script-trace ---> Da info de como va elscript

--script-updatedb

**Para usar un script solo hay que poner: namp --script Nombre\_del\_script objetivo** --> Al poner el script se ejecutará tanto el script como el escaner, asi que tambien se pueden poner opciones del escaner, podemos añadir **“safe=1”** para que se ejecuten solo los que sean seguros.

**Control tiempo**

**Nmap puede modificar el tiempo en segundos, minutos, ms:** --host-timeout arguments 900000ms, 900, 900s, and 15m all do the same thing.

Nmap divide el numero total de host a escanear en grupos y analiza esos grupos en bloques de forma que hasta que no han sido analizados todos, no pasa al siguiente bloque (y el usuario tampoco recibe ninguna actualización hasta que se haya analizado el bloque) de esta forma, es más óptimo para nmap usar grupos grandes. Por defecto en clase C usa 256.

Se puede cambiar con\*\*--min-hostgroup\*\* ***<numhosts>*;** **--max-hostgroup** ***<numhosts>*** (Adjust parallel scan group sizes)

Se puede controlar el numero de escaners en paralelo pero es mejor que no (nmpa ya incorpora control automatico en base al estado de la red): **--min-parallelism** ***<numprobes>*;** **--max-parallelism** ***<numprobes>***

Podemos modificar el rtt timeout, pero no suele ser necesario: **--min-rtt-timeout** ***<time>*,** **--max-rtt-timeout** ***<time>*,** **--initial-rtt-timeout** ***<time>***

Podemos modificar el numero de intentos:**--max-retries** ***<numtries>***

Podemos modificar el tiempo de escaneado de un host: **--host-timeout** ***<time>***

Podemos modificar el tiempo entre cada prueba para que vaya despacio: **--scan-delay** ***<time>*;** **--max-scan-delay** ***<time>***

Podemos modificar el numero de paquetes por segundo: **--min-rate** ***<number>*;** **--max-rate** ***<number>***

Muchos puertos tardan mucho en responder al estar filtrados o cerrados, si solo nos interesan los abiertos, podemos ir más rápido con: **--defeat-rst-ratelimit**

Para definir lo agresivo que queremos que sea nmap: -T paranoid|sneaky|polite|normal|aggressive|insane

-T (0-1)

-T0 --> Solo se escanea 1 puerto a la vez y se espera 5min hasta el siguiente

-T1 y T2 --> Muy parecidos pero solo esperan 15 y 0,4seg respectivamente enttre cada prueba

-T3 --> Funcionamiento por defecto, incluye en paralelo

-T4 --> --max-rtt-timeout 1250ms --min-rtt-timeout 100ms --initial-rtt-timeout 500ms --max-retries 6 --max-scan-delay 10ms

-T5 --> --max-rtt-timeout 300ms --min-rtt-timeout 50ms --initial-rtt-timeout 250ms --max-retries 2 --host-timeout 15m --max-scan-delay 5ms

**Firewall/IDS**

No dejan pasar a puertos y analizan paquetes.

**-f** Para fragmentar paquetes, por defecto los fragmenta en 8bytes después de la cabecera, para especificar ese tamaño usamos ..mtu (con esto, no usar -f), el offset debe ser multiplo de 8. **Escaners de version y scripts no soportan la fragmentacion**

**-D decoy1,decoy2,ME** Nmap envia escaneres pero con otras direcciones IPs como origen, de esta forma te esconden a ti. Si pones el ME en la lista, nmap te situara ahi, mejor poner 5 o 6 antes de ti para que te enmascaren completamente. Se pueden generar iPs aleatorias con RND:<numero> Para generar <numero> de Ips aleatorias. No funcionan con detector de versiones sin conexion de TCP. Si estas dentro de una red, te interesa usar Ips que esten activas, pues sino será muy facil averiguar que tu eres la unica activa.

Para usar Ips aleatorias: nmap-D RND: 10 Ip\_objetivo

**-S IP** Para cuando Nmap no pilla tu dirección Ip se la tienes que dar con eso. También sirve para hacer pensar que hay otro objetivo escaneandoles.

**-e <interface>** Para elegir la interfaz

Muchos administradores dejan puertos de entrada abiertos para que todo funcione correctamente y les es más fácil que buscar otra solución. Estos pueden ser los puertos DNS o los de FTP... para busca esta vulnerabilidad nmap incorpora: **--source-port** ***<portnumber>*;-g** ***<portnumber>*** *Son equivalentes*

**--data** ***<hex string>*** Para enviar texto hexadecimal: --data 0xdeadbeef and --data \xCA\xFE\x09

**--data-string** ***<string>*** Para enviar un texto normal: --data-string "Scan conducted by Security Ops, extension 7192"

**--data-length** ***<number>*** Nmap envía solo cabeceras, con esto logramos que añada a estar un numero de bytes mas (que se generaran aleatoriamente)

Para configurar el paquete IP completamente usar **--ip-options**

If you wish to see the options in packets sent and received, specify --packet-trace. For more information and examples of using IP options with Nmap, see <http://seclists.org/nmap-dev/2006/q3/52>.

**--ttl** ***<value>***

**--randomize-hosts** Para que el ataque sea menos obvio

**--spoof-mac** ***<MAC address, prefix, or vendor name>*** Para cambiar la mac ejemplos: Apple, 0, 01:02:03:04:05:06, deadbeefcafe, 0020F2, and Cisco

**--proxies** ***<Comma-separated list of proxy URLs>*** Para usar proxies, a veces un proxy no mantiene tantas conexiones abiertas como nmap quiere por lo que habria que modificar el paralelismo: --max-parallelism

**-sP** Para descubrir host en la red en la que estamos por ARP

Muchos administradores crean una regla en el firewall que permite pasar todos los paquetes que provienen de un puerto en particular (como el 20,53 y 67), podemos decire a nmap que mande nuestros paquetes desde esos puertos: **nmap --source-port 53 Ip**

**Salidas**

**-oN file** Salida normal

**-oX file** Salida XML

**-oS file** Salida de script kidies

**-oG file** Salida grepable

**-oA file** Todos menos -oS

**-v level** verbosity

**-d level** debugin

**--reason** Porqué del host y estado

**--stats-every time** Cada ese tiempo nos dice como va

**--packet-trace** Para ver que paquetes salen se pueden especificar filtros como: --version-trace o --script-trace

**--open** muestra los abiertos, abiertos|filtrados y los no filtrados

**--resume file** Saca un resumen

**Miscelanea**

**-6** Permite ipv6

**-A** es lo mismo que -O -sV -sC --traceroute

**Run time**

Mientras corre nmap podemos cambiar opciones:

v / V Increase / decrease the verbosity level

d / D Increase / decrease the debugging Level

p / P Turn on / off packet tracing

? Print a runtime interaction help screen

**Vulscan**

Script de nmap que mira las versiones de los servicios obtenidos en una base de datos offline (que descarga de otras muy importantes) y devuelve las posibles vulnerabilidades

Las BD que usa son:

1. Scipvuldb.csv | <http://www.scip.ch/en/?vuldb>
2. Cve.csv | [http://cve.mitre.org](http://cve.mitre.org/)
3. Osvdb.csv | [http://www.osvdb.org](http://www.osvdb.org/)
4. Securityfocus.csv | <http://www.securityfocus.com/bid/>
5. Securitytracker.csv | [http://www.securitytracker.com](http://www.securitytracker.com/)
6. Xforce.csv | [http://xforce.iss.net](http://xforce.iss.net/)
7. Exploitdb.csv | [http://www.exploit-db.com](http://www.exploit-db.com/)
8. Openvas.csv | [http://www.openvas.org](http://www.openvas.org/)

Para descargarlo e instalarlo en la carpeta de Nmap:

wget http://www.computec.ch/projekte/vulscan/download/nmap\_nse\_vulscan-2.0.tar.gz && tar -czvf nmap\_nse\_vulscan-2.0.tar.gz vulscan/ && sudo cp -r vulscan/ /usr/share/nmap/scripts/

También habría que descargar los paquetes de las BD y añadirlos a /usr/share/nmap/scripts/vulscan/

Uso:

Para usar todos: sudo nmap -sV --script=vulscan HOST\_A\_ESCANEAR

Para usar una BD específica: sudo nmap -sV --script=vulscan --script-args vulscandb=cve.csv HOST\_A\_ESCANEAR

## Speed Up Nmap Service scan x16

According [**to this post**](https://joshua.hu/nmap-speedup-service-scanning-16x) you can speed up the nmap service analysis by modifying all the **totalwaitms** values in **/usr/share/nmap/nmap-service-probes** to **300** and **tcpwrappedms** to **200**.

Moreover, probes which do not have a specifically defined **servicewaitms** use a default value of **5000**. Therefore, we can either add values to each of the probes, or we can **compile nmap** ourselves and change the default value in [**service\_scan.h**](https://github.com/nmap/nmap/blob/master/service_scan.h#L79).

If you don't want to change the values of **totalwaitms** and **tcpwrappedms** at all in the /usr/share/nmap/nmap-service-probes file, you can edit the [parsing code](https://github.com/nmap/nmap/blob/master/service_scan.cc#L1358) such that these values in the nmap-service-probes file are completely ignored.

**Network Protocols Explained (ESP)**

### Multicast DNS (mDNS)

The **mDNS** protocol is designed for IP address resolution within small, local networks without a dedicated name server. It operates by multicasting a query within the subnet, prompting the host with the specified name to respond with its IP address. All devices in the subnet can then update their mDNS caches with this information.

Key points to note:

* **Domain Name Relinquishment**: A host can release its domain name by sending a packet with a TTL of zero.
* **Usage Restriction**: mDNS typically resolves names ending in **.local** only. Conflicts with non-mDNS hosts in this domain require network configuration adjustments.
* **Networking Details**:
  + Ethernet multicast MAC addresses: IPv4 - 01:00:5E:00:00:FB, IPv6 - 33:33:00:00:00:FB.
  + IP addresses: IPv4 - 224.0.0.251, IPv6 - ff02::fb.
  + Operates over UDP port 5353.
  + mDNS queries are confined to the local network and do not cross routers.

### DNS-SD (Service Discovery)

DNS-SD is a protocol for discovering services on a network by querying specific domain names (e.g., \_printers.\_tcp.local). A response includes all related domains, such as available printers in this case. A comprehensive list of service types can be found [here](http://www.dns-sd.org/ServiceTypes.html).

### SSDP (Simple Service Discovery Protocol)

SSDP facilitates the discovery of network services and is primarily utilized by UPnP. It's a text-based protocol using UDP over port 1900, with multicast addressing. For IPv4, the designated multicast address is 239.255.255.250. SSDP's foundation is [HTTPU](https://en.wikipedia.org/wiki/HTTPU), an extension of HTTP for UDP.

### Web Service for Devices (WSD)

Devices connected to a network can identify available services, like printers, through the Web Service for Devices (WSD). This involves broadcasting UDP packets. Devices seeking services send requests, while service providers announce their offerings.

### OAuth 2.0

OAuth 2.0 is a protocol facilitating secure, selective sharing of user information between services. For instance, it enables services to access user data from Google without multiple logins. The process involves user authentication, authorization by the user, and token generation by Google, allowing service access to the specified user data.

### RADIUS

RADIUS (Remote Authentication Dial-In User Service) is a network access protocol primarily used by ISPs. It supports authentication, authorization, and accounting. User credentials are verified by a RADIUS server, potentially including network address verification for added security. Post-authentication, users receive network access and their session details are tracked for billing and statistical purposes.

### SMB and NetBIOS

#### SMB (Server Message Block)

SMB is a protocol for sharing files, printers, and ports. It operates directly over TCP (port 445) or via NetBIOS over TCP (ports 137, 138). This dual compatibility enhances connectivity with various devices.

#### NetBIOS (Network Basic Input/Output System)

NetBIOS manages network sessions and connections for resource sharing. It supports unique names for devices and group names for multiple devices, enabling targeted or broadcast messaging. Communication can be connectionless (no acknowledgment) or connection-oriented (session-based). While NetBIOS traditionally operates over protocols like IPC/IPX, it's commonly used over TCP/IP. NetBEUI, an associated protocol, is known for its speed but was also quite verbose due to broadcasting.

### LDAP (Lightweight Directory Access Protocol)

LDAP is a protocol enabling the management and access of directory information over TCP/IP. It supports various operations for querying and modifying directory information. Predominantly, it's utilized for accessing and maintaining distributed directory information services, allowing interaction with databases designed for LDAP communication.

### Active Directory (AD)

Active Directory is a network-accessible database containing objects like users, groups, privileges, and resources, facilitating centralized management of network entities. AD organizes its data into a hierarchical structure of domains, which can encompass servers, groups, and users. Subdomains allow further segmentation, each potentially maintaining its own server and user base. This structure centralizes user management, granting or restricting access to network resources. Queries can be made to retrieve specific information, like contact details, or to locate resources, like printers, within the domain.