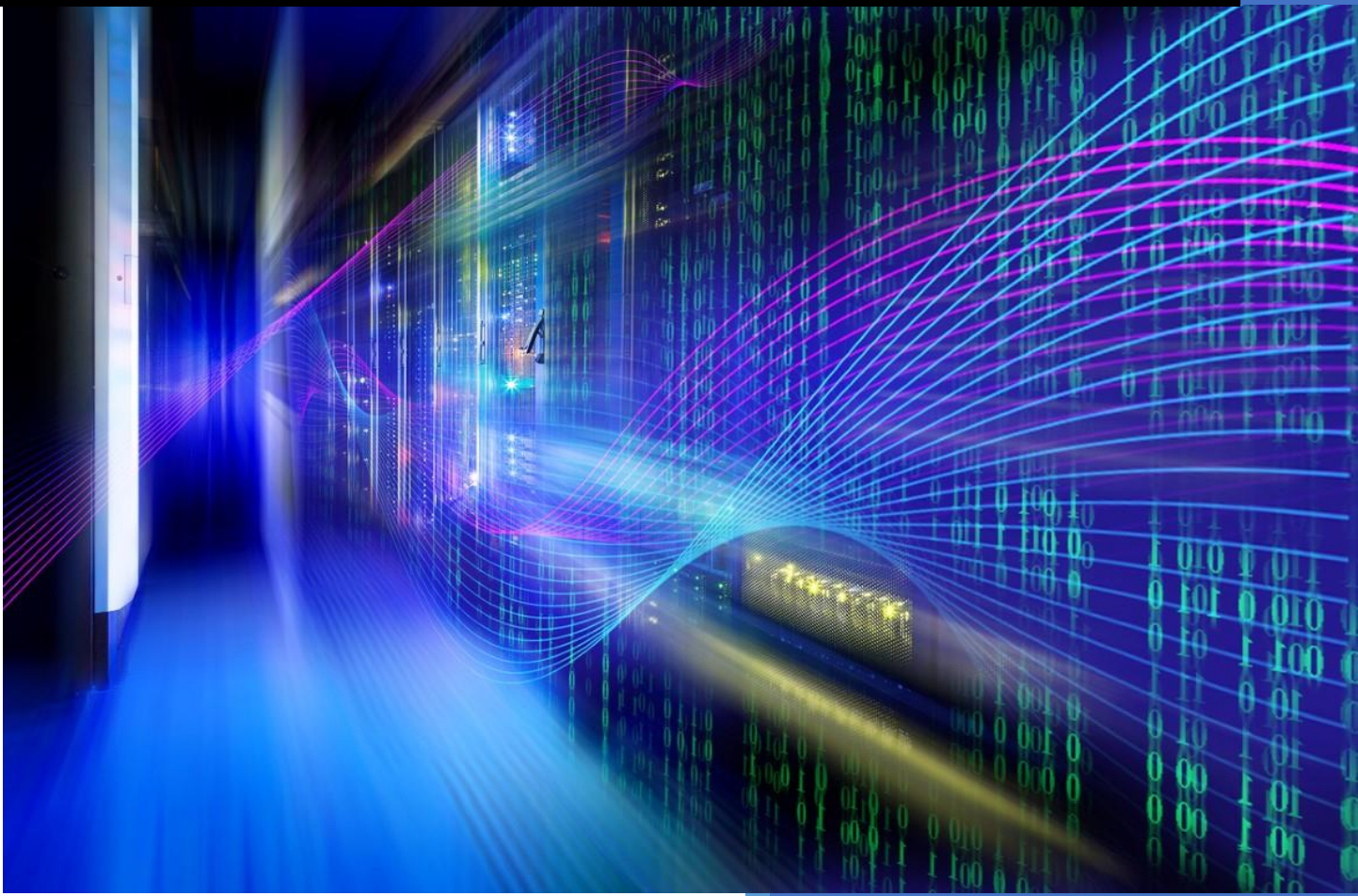


Actividad Integral Evaluable

Sistemas Informáticos



David Pires

**1º Desarrollo Aplicaciones
Multiplataforma**

1. Introducción a la actividad.	3
1.1. Objetivo.	3
1.2. Ámbito de aplicación.	3
2. Actividades modulares network.	3
2.1. Módulo Network 1: Script de Host activos.	3
2.1.1. Objetivo.	3
2.1.2. Herramientas necesarias.	3
2.1.3. Proceso de descarga de las herramientas necesarias.	4
2.1.4. Realización de la actividad.	4
1) Buscar la red de nuestro equipo.	5
2) Realización de un ping.	6
3) Creación de script con bucle for.	6
4) Modificación del bucle for con variable “%%i”.	7
5) Utilización de for anidado para búsqueda de todos los equipos de todas las subredes.	9
6) Uso del findstr para mostrar sólo los equipos vivos.	9
7) Implementación del script en Python.	9
2.2. Módulo Network 2: Calculadora de direcciones IP en binario.	14
2.2.1. Objetivo.	14
2.2.2. Herramientas necesarias.	14
2.2.3. Proceso de descarga de las herramientas necesarias.	14
2.2.4. Realización de la actividad.	14
1) Realizar el cálculo de la red.	14
2) Bits relevantes.	14
3) Realización del script.	15
4) Implementación de un contador en el script.	16
5) Exportación de datos a txt.	18
2.3. Módulo Network 3: Sniffer de captura de tráfico de red.	18
2.3.1. Objetivo.	18
2.3.2. Herramientas necesarias.	18
2.3.3. Proceso de descarga de las herramientas necesarias.	18
2.3.4. Realización de la actividad.	19
1) Primeros pasos en la aplicación.	19
2) Aplicación de filtros de búsqueda.	21
3) Resolución de los apartados pedidos.	21
2.4. Módulo Network 4: Network Scanner, escaneador de red.	24

2.4.1. Objetivo.....	24
2.4.2. Herramientas necesarias.....	24
2.4.3. Proceso de descarga de las herramientas necesarias.	24
2.4.4. Realización de la actividad.	25
1) Equipos visibles a nivel de IP:.....	25
2) Listado de puertos TCP y UDP de los equipos de la red:	26
3) Servicio más probable en ejecución en cada puerto en base a IANA.....	28
3. Conclusiones del trabajo.....	30
4. Tabla resumen de imágenes.....	32

1. Introducción a la actividad.

1.1. Objetivo.

Las herramientas de seguridad informática son esenciales para la protección de sistemas y redes que se utilizan a diario en el mundo.

Los scripts de escaneo, los programas de sniffer y los programas para detectar vulnerabilidades son tres tipos de herramientas fundamentales que permiten identificar y mitigar riesgos de seguridad. La elección de la herramienta adecuada dependerá de las necesidades específicas de cada organización.

Durante este trabajo, se pondrá el foco en distintos programas que están destinados, algunos parcialmente y otros completamente, a la detección de posibles vulnerabilidades en los equipos informáticos.

Además, se utilizará la Inteligencia Artificial, una de las herramientas informáticas más potentes de los últimos tiempos para poder solventar los distintos problemas que se vayan interponiendo en el transcurso de la práctica.

1.2. Ámbito de aplicación.

Para este trabajo se utilizará el equipo personal del alumno, y el ámbito de aplicación de las herramientas solicitadas será la red de ordenadores del centro, con la que se obtendrán los datos necesarios para pasar la práctica satisfactoriamente.

2. Actividades modulares network.

2.1. Módulo Network 1: Script de Host activos.

2.1.1. Objetivo.

El objetivo de esta práctica es la creación de un script que identifique qué equipos “están vivos” dentro de la red a la que pertenece el equipo del alumno.

2.1.2. Herramientas necesarias.

Para la creación del script, que se va a desarrollar a través del lenguaje de programación *Python*, se van a utilizar los siguientes programas:

- La consola de Windows CMD.

- Python: Uno de los lenguajes de programación más importantes del momento, ya que en los últimos años ha aumentado mucho su popularidad, debido a su eficiencia y facilidad.
- PyCharm: Este programa es un IDE (Integrated Development Environment) de Python, que simplifica enormemente la programación y el proceso de desarrollo, ya que es una interfaz muy cómoda y limpia para cualquier desarrollador de Python.

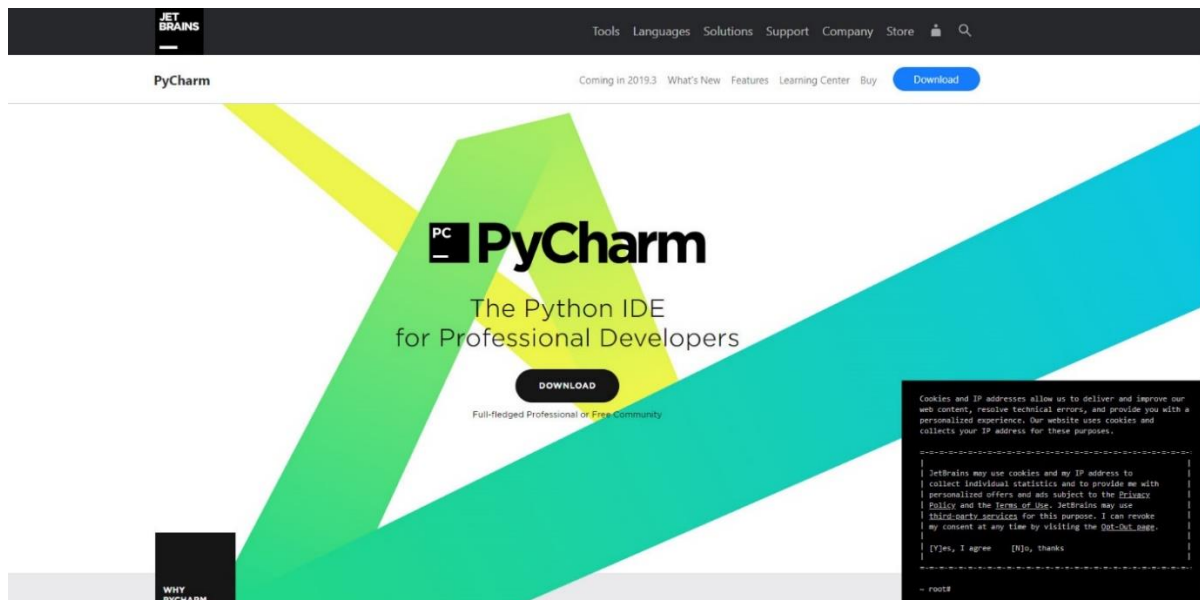


Imagen 1. Página oficial de PyCharm

- Google Bard: Sistema de IA (Inteligencia Artificial) creado por Google, que no es más que un sistema conversacional con el que se puede interactuar de manera directa para resolver cualquier pregunta o duda del usuario. Actualmente es uno de los más utilizados, por detrás de ChatGPT.

2.1.3. Proceso de descarga de las herramientas necesarias.

En este caso, tanto la CMD (que viene integrada en el sistema de Windows), el PyCharm y la extensión de Python están ya disponibles para ser utilizadas en el equipo, por lo que en este caso no se requiere de ningún proceso de instalación.

2.1.4. Realización de la actividad.

Antes de comenzar a utilizar la IA para poder hacer un programa funcional en Python, se creará el script necesario para saber, como se decía en el objetivo de la práctica, qué equipos están vivos en la red, para lo cual debemos responder a la siguiente pregunta: **¿Qué significa que un equipo “está vivo” en una red? -->**

La respuesta es sencilla, ya que un equipo vivo en una red es accesible desde otro equipo de esa red, y para saber si se puede acceder o no, se utiliza una herramienta de diagnóstico llamada **ping**.

El ping funciona a través de un protocolo llamado **ICMP** (Internet Control Message Protocol), con el cual envía paquetes de datos a otro dispositivo. Si recibe respuesta, significará que el dispositivo objetivo está en la misma red, es decir, que está vivo.

A partir de este punto, se estructurará el proceso paso a paso y adjuntando capturas de pantalla de todo.

1) *Buscar la red de nuestro equipo.*

Para conocer la red del equipo que se está se utiliza el comando **ipconfig**, con el cual se pueden conocer todos los valores de configuración de red TCP/IP actuales.

```
Adaptador de Ethernet vEthernet (Default Switch):

    Sufijo DNS específico para la conexión. . . : 
    Vínculo: dirección IPv6 local. . . : fe80::e33a:c971:f24e:877f%54
    Dirección IPv4. . . . . : 172.27.240.1
    Máscara de subred . . . . . : 255.255.240.0
    Puerta de enlace predeterminada . . . . . : 

Adaptador de Ethernet Ethernet:

    Sufijo DNS específico para la conexión. . . : 
    Vínculo: dirección IPv6 local. . . : fe80::25ea:eddd:42b7:5652%9
    Dirección IPv4. . . . . : 192.168.128.115
    Máscara de subred . . . . . : 255.255.252.0
    Puerta de enlace predeterminada . . . . . : 192.168.128.1

Adaptador de Ethernet VirtualBox Host-Only Network:

    Sufijo DNS específico para la conexión. . . : 
    Vínculo: dirección IPv6 local. . . : fe80::8856:5b64:d45a:3127%17
    Dirección IPv4. . . . . : 192.168.56.1
    Máscara de subred . . . . . : 255.255.255.0
    Puerta de enlace predeterminada . . . . . : 

Adaptador de LAN inalámbrica Wi-Fi:

    Estado de los medios. . . . . : medios desconectados
    Sufijo DNS específico para la conexión. . . : 

Adaptador de LAN inalámbrica Conexión de área local* 1:

    Estado de los medios. . . . . : medios desconectados
    Sufijo DNS específico para la conexión. . . :
```

Imagen 2. *Ipconfig del equipo del alumno*

En el apartado de Adaptador de Ethernet se encuentra la dirección ip del equipo, además de su máscara de red.

Ahora vamos a estudiar rápidamente la ip y la máscara, para poder conocer el rango de aplicación necesario del script.

Observando ambas, se puede comprobar que tanto el octeto 192 como el 168 de la ip forman parte de la red, ya que, si se enfrenta en binario su valor frente al de la máscara en los mismos octetos, se obtiene el mismo resultado:

192 AND 255 (en binario sería igual a) = 11000000 AND 11111111 = 11000000 = 192;

168 AND 255 (en binario sería igual a) = 10101000 AND 11111111 = 10101000 = 168;

Sin embargo, en los siguientes dos octetos sucede algo distinto, ya que la máscara para el tercer octeto es 252. Esto significa que los valores del tercer octeto para esta ip, se van a mover entre:

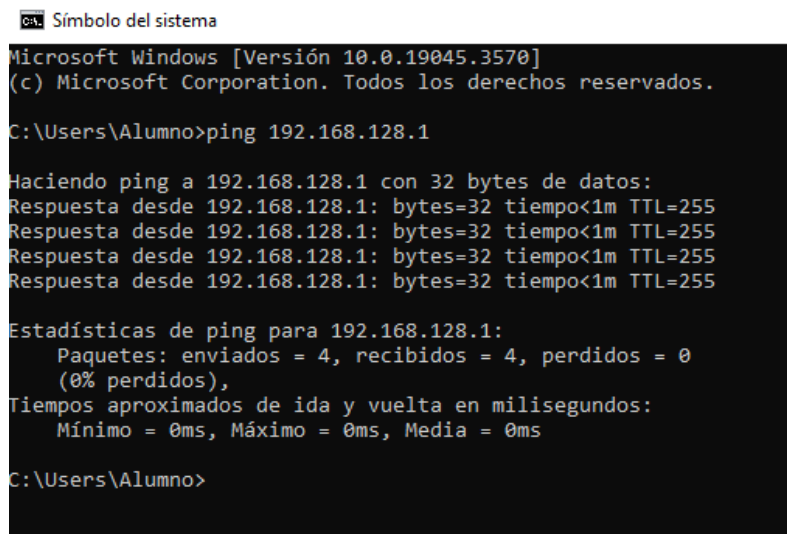
- 128 / 252 -- 129 / 253 -- 130 / 254 -- 131 / 255

Con esto, se debe considerar que esas subredes con sus respectivos equipos forman parte de la misma red que el ordenador del alumno, por lo que se deben considerar para la realización del script.

Por último, el último octeto de la ip será el identificador de cada equipo de cada una de las subredes de la red.

2) Realización de un ping.

Lanzamos el comando escribiendo en la consola de Windows “ping”, seguido de una dirección ip. Para asegurarnos de que obtenemos respuesta, se realiza el envío de datos a la ip 192.168.128.1, ya que normalmente es la salida predeterminada del router de nuestra red.



```

C:\> Símbolo del sistema

Microsoft Windows [Versión 10.0.19045.3570]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\Alumno>ping 192.168.128.1

Haciendo ping a 192.168.128.1 con 32 bytes de datos:
Respuesta desde 192.168.128.1: bytes=32 tiempo<1m TTL=255
Respuesta desde 192.168.128.1: bytes=32 tiempo<1m TTL=255
Respuesta desde 192.168.128.1: bytes=32 tiempo<1m TTL=255
Respuesta desde 192.168.128.1: bytes=32 tiempo<1m TTL=255

Estadísticas de ping para 192.168.128.1:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
              (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 0ms, Máximo = 0ms, Media = 0ms

C:\Users\Alumno>
```

Imagen 3. Ping a la IP de la ruta predeterminada

3) Creación de script con bucle for.

Ya que el ping es un comando que se lanza a una sola dirección ip a la vez en su formato estándar, se debe encontrar la manera de poder realizar tantos pings como equipos tenga la red.

Para ello, se plantea el uso de un bucle for, con el que se podrá lanzar el ping tantas veces como marque el bucle.

```
for %%i in (1,2,3) do (  
ping 192.168.128.1  
)
```

Imagen 4. Bucle for para realizar 3 veces un ping

```
D:\Seguridad>escaneo  
D:\Seguridad>for %i in (1 2 3) do (ping 192.168.128.1 )  
D:\Seguridad>(ping 192.168.128.1 )  
Haciendo ping a 192.168.128.1 con 32 bytes de datos:  
Respuesta desde 192.168.128.1: bytes=32 tiempo<1m TTL=255  
Respuesta desde 192.168.128.1: bytes=32 tiempo<1m TTL=255  
Respuesta desde 192.168.128.1: bytes=32 tiempo<1m TTL=255  
Respuesta desde 192.168.128.1: bytes=32 tiempo<1m TTL=255  
Estadísticas de ping para 192.168.128.1:  
Paquetes: enviados = 4, recibidos = 4, perdidos = 0  
(0% perdidos),  
Tiempos aproximados de ida y vuelta en milisegundos:  
Mínimo = 0ms, Máximo = 0ms, Media = 0ms  
D:\Seguridad>(ping 192.168.128.1 )  
Haciendo ping a 192.168.128.1 con 32 bytes de datos:  
Respuesta desde 192.168.128.1: bytes=32 tiempo<1m TTL=255  
Respuesta desde 192.168.128.1: bytes=32 tiempo<1m TTL=255  
Respuesta desde 192.168.128.1: bytes=32 tiempo<1m TTL=255  
Respuesta desde 192.168.128.1: bytes=32 tiempo<1m TTL=255  
Estadísticas de ping para 192.168.128.1:  
Paquetes: enviados = 4, recibidos = 4, perdidos = 0  
(0% perdidos),  
Tiempos aproximados de ida y vuelta en milisegundos:  
Mínimo = 0ms, Máximo = 0ms, Media = 0ms  
D:\Seguridad>(ping 192.168.128.1 )  
Haciendo ping a 192.168.128.1 con 32 bytes de datos:  
Respuesta desde 192.168.128.1: bytes=32 tiempo<1m TTL=255  
Respuesta desde 192.168.128.1: bytes=32 tiempo<1m TTL=255  
Respuesta desde 192.168.128.1: bytes=32 tiempo<1m TTL=255  
Respuesta desde 192.168.128.1: bytes=32 tiempo<1m TTL=255  
Estadísticas de ping para 192.168.128.1:  
Paquetes: enviados = 4, recibidos = 4, perdidos = 0  
(0% perdidos),  
Tiempos aproximados de ida y vuelta en milisegundos:  
Mínimo = 0ms, Máximo = 0ms, Media = 0ms
```

Imagen 5. Ejecución del bucle for para 3 pings

En este for se representa un bucle en el cual lanza un paquete ping 3 veces a la dirección de ruta predeterminada del router. Por lo tanto, sabiendo que el contenido del paréntesis tras el “in” permite crear un rango, se puede modificar el script para que en vez de realizar el bucle un número de veces establecidas, lo realice en un rango determinado a través de la definición de una variable.

4) Modificación del bucle for con variable “%%i”.

Editamos el script de escaneo, y le damos un valor al último octeto con un %%i, para que recorra las redes acabadas en 1, 2 y 3.

```
for %%i in (1,2,3) do (  
ping 192.168.128.%%i  
)
```

Imagen 6. Bucle for para las ips acabadas entre 1 y 3


```

D:\Seguridad>escaneo

D:\Seguridad>for %i in (1 2 3 4 5) do (ping 192.168.128.%i -n 1 )

D:\Seguridad>(ping 192.168.128.1 -n 1 )

Haciendo ping a 192.168.128.1 con 32 bytes de datos:
Respuesta desde 192.168.128.1: bytes=32 tiempo<1m TTL=255

Estadísticas de ping para 192.168.128.1:
    Paquetes: enviados = 1, recibidos = 1, perdidos = 0
              (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 0ms, Máximo = 0ms, Media = 0ms

D:\Seguridad>(ping 192.168.128.2 -n 1 )

Haciendo ping a 192.168.128.2 con 32 bytes de datos:
Respuesta desde 192.168.128.115: Host de destino inaccesible.

Estadísticas de ping para 192.168.128.2:
    Paquetes: enviados = 1, recibidos = 1, perdidos = 0
              (0% perdidos),

D:\Seguridad>(ping 192.168.128.3 -n 1 )

Haciendo ping a 192.168.128.3 con 32 bytes de datos:
Respuesta desde 192.168.128.115: Host de destino inaccesible.

Estadísticas de ping para 192.168.128.3:
    Paquetes: enviados = 1, recibidos = 1, perdidos = 0
              (0% perdidos),

D:\Seguridad>(ping 192.168.128.4 -n 1 )

Haciendo ping a 192.168.128.4 con 32 bytes de datos:
Respuesta desde 192.168.128.115: Host de destino inaccesible.

Estadísticas de ping para 192.168.128.4:
    Paquetes: enviados = 1, recibidos = 1, perdidos = 0
              (0% perdidos),

D:\Seguridad>(ping 192.168.128.5 -n 1 )

Haciendo ping a 192.168.128.5 con 32 bytes de datos:
Respuesta desde 192.168.128.115: Host de destino inaccesible.

Estadísticas de ping para 192.168.128.5:
    Paquetes: enviados = 1, recibidos = 1, perdidos = 0
              (0% perdidos),

```

Imagen 7. Ejecución del bucle for para ping a Ips acabadas entre 1 y 3

Además, se puede modificar el rango de la condición del for, pidiéndole que establezca un rango del 1 al 255, pasando de uno en uno por todos los equipos. Esto se realiza a través del comando “/l”, y, además, se añade el comando “-n 1”, con el cual se pueden establecer cuantas respuestas se reciben al ping. De manera estándar este comando envía 4 paquetes y recibe 4 respuestas, pero con este filtro sólo se obtendrá una respuesta, la cual es suficiente para conocer si el equipo es accesible o no.

```

for /l %i in (1, 1 ,255) do (
ping 192.168.128.%i -n 1
)|

```

Imagen 8. Bucle for para todas las Ips entre la 1 y la 255

5) Utilización de for anidado para búsqueda de todos los equipos de todas las subredes.

Como se ha dicho antes, los equipos de la red dependen del último octeto, sino que hay un rango de subredes dentro de la misma red, que también se deben cubrir con este script.

Por tanto, si se utiliza la misma estructura de for usada para la localización de equipos de la subred 128, y se aplica una variable “%%j” en ese lugar del octeto, se obtendría un script de escaneo de todos los equipos de la red al completo

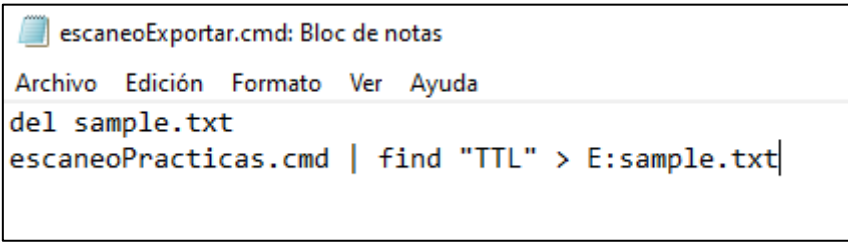
```
for /l %%j in (128, 1 ,131) do (  
    for /l %%i in (1, 1 ,255) do (  
        ping 192.168.%%j.%%i -n 1  
    )  
)
```

Imagen 9. Bucle for anidado para todo el rango de equipos de la red.

6) Uso del findstr para mostrar sólo los equipos vivos.

Aunque ya está compuesto el script, se debe considerar que, aunque solo se genere una respuesta, será tanto de equipos vivos como no accesibles, por lo que se establecerá un último filtrado para conocer los que responden al ping.

Para ello, se llama al script desde otro txt, y se aplica el | find TTL. Además, se va a dejar preparada función, a través de la cual crea un archivo txt llamado sample.txt en el que se exportarán los resultados.



```
escaneoExportar.cmd: Bloc de notas  
Archivo Edición Formato Ver Ayuda  
del sample.txt  
escaneoPracticas.cmd | find "TTL" > E:sample.txt
```

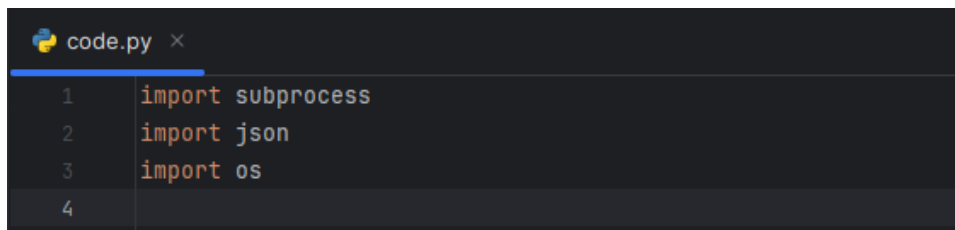
Imagen 10. Find TTL del script de escaneo

7) Implementación del script en Python.

Ya que Python es una herramienta potente, permitirá crear un script funcional y en el que el usuario elija entre varias opciones.

7.1)

Primero, se importa la función json y el subprocess para la parte de exportar los resultados, y la función os para poder interactuar con el sistema operativo de Windows.



```
code.py x
1 import subprocess
2 import json
3 import os
4
```

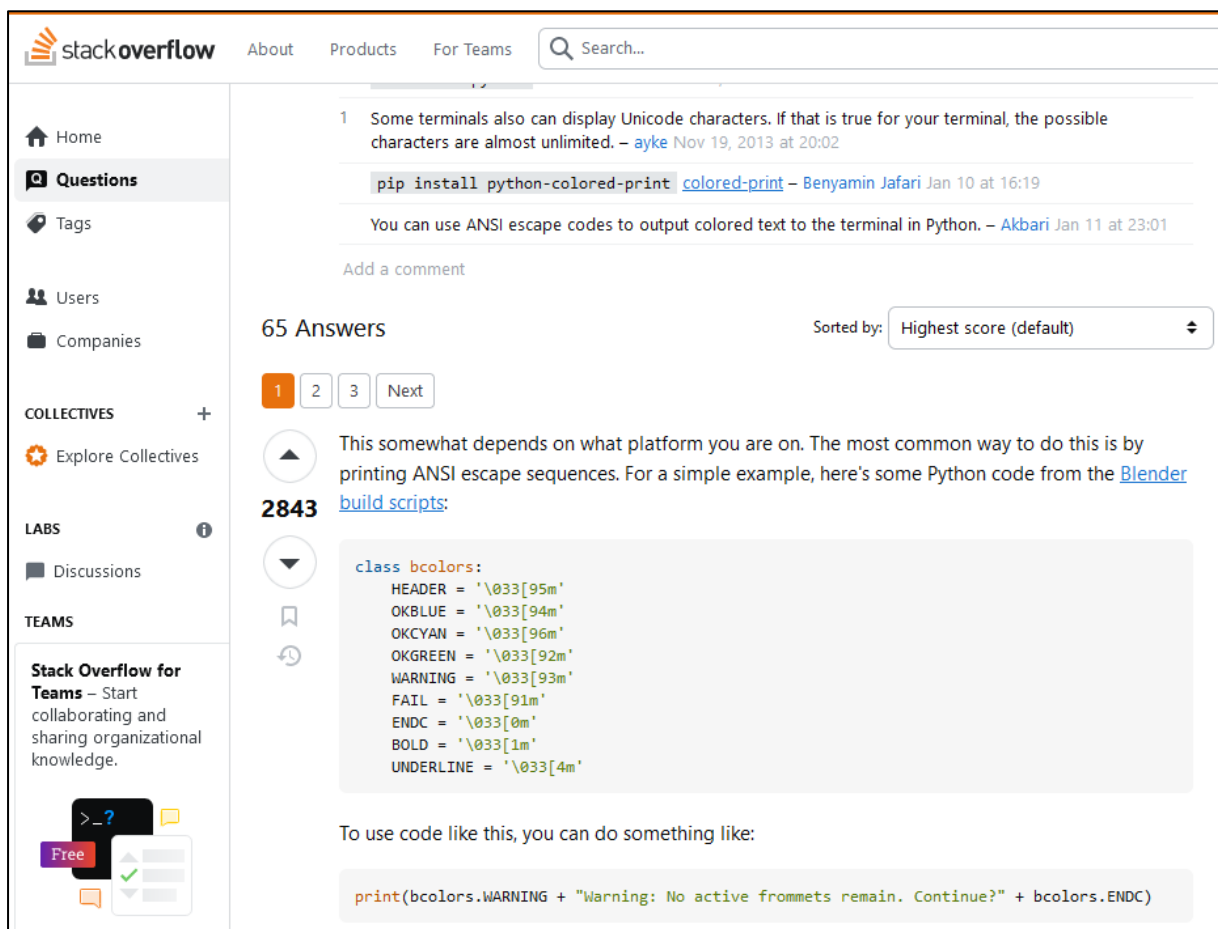
Imagen 11. Imports de Python

7.2)

Para darle un formato más agradable a la vista, se realizó una búsqueda en el navegador para buscar opciones con las que se pudieran añadir colores a los resultados de los escáneres por pantalla.

En la página [Stack Overflow](https://stackoverflow.com) se encontró una clase ya creada en la que se establecía un tag a distintos colores, que luego se podían llamar dentro de un String.

Esta clase se implementa de manera literal en el código propio.



The screenshot shows a Stack Overflow page for a question about displaying Unicode characters in a terminal. The question asks for a way to output colored text to the terminal in Python. The top answer, by Benjamin Jafari, provides a class named `bcolors` that defines ANSI escape codes for various colors and styles. The code is as follows:

```
class bcolors:
    HEADER = '\033[95m'
    OKBLUE = '\033[94m'
    OKCYAN = '\033[96m'
    OKGREEN = '\033[92m'
    WARNING = '\033[93m'
    FAIL = '\033[91m'
    ENDC = '\033[0m'
    BOLD = '\033[1m'
    UNDERLINE = '\033[4m'
```

The answer also includes a sample usage: `print(bcolors.WARNING + "Warning: No active frommets remain. Continue?" + bcolors.ENDC)`.

```

code.py x
1  import subprocess
2  import json
3  import os
4  |
4 usages
5  class bcolors:
6      HEADER = '\033[95m'
7      OKBLUE = '\033[94m'
8      OKCYAN = '\033[96m'
9      OKGREEN = '\033[92m'
10     WARNING = '\033[93m'
11     FAIL = '\033[91m'
12     ENDC = '\033[0m'
13     BOLD = '\033[1m'
14     UNDERLINE = '\033[4m'
15

```

Imagen 12. Clase bcolors obtenida en StackOverflow

7.3)

Se pide a Google Bard que implemente en el código un menú, a través del cual el usuario pueda elegir entre ejecutar el código, leer la documentación sobre el script, exportar los resultados o salir del programa.



Aquí está el código para un menú que permite al usuario elegir entre ejecutar el código, ver un texto con documentación o salir:

Python

```

import subprocess

def mostrar_menu():
    print("""
    print("      (_)"
          (oo)
    /-----\
    * / |  ||
      ~ ~ ~ ~ ~
    print("""
    print("Bienvenido al script de ping.")
    print("¿Qué desea hacer?")
    print("1. Ejecutar el código")
    print("2. Acceder a la documentación")
    print("3. Salir")

def obtener_opcion():
    opcion = input("Introduzca una opción: ")
    while opcion not in ("1", "2", "3"):
        print("Opción no válida.")
        opcion = input("Introduzca una opción: ")
    return opcion

```

```

mostrar_menu()
opcion = obtener_opcion()

while opcion != "3":
    if opcion == "1":
        ejecutar_codigo()
    elif opcion == "2":
        acceder_a_documentacion()
    mostrar_menu()
    opcion = obtener_opcion()

print("\nHasta la próxima!")

```

Imagen 13. Implementación de menú con Google Bard

Debido a que el ASCII mostrado en el bard no funciona, se decide descartarlo y mantener sólo el menú.

7.4)

A través de distintas búsquedas en bard y de varios intentos de pasar el código de bash a Python, se consigue un código funcional para realizar el ping, pero sólo a los equipos de la subred 128:

```

1 usage
43 def ejecutar_codigo():
44     """
45     Ejecuta el código del script.
46     """
47     ip = "192.168.128"
48     print("Escaneando las IP's desde la : " + ip + ".1 hasta la " + ip + ".255")
49
50     subnet = ip.split(".")
51
52     FNULL = open(os.devnull, 'w')
53
54     for x in range(1, 255):
55         ip2 = subnet[0] + "." + subnet[1] + "." + subnet[2] + "." + str(x)
56         response = subprocess.Popen(["ping", "-n", "1", ip2], stdout=FNULL, stderr=subprocess.STDOUT).wait()
57         if response == 0:
58             print(bcolors.OKGREEN + ip2, 'está viva!' + bcolors.ENDC)
59         else:
60             print(bcolors.FAIL + ip2, 'está muerta!' + bcolors.FAIL)
61

```

Imagen 14. Script de ping en Python

En este script, se llama a la clase bcolors para que dé como resultado en verde que la ip está viva y en rojo que está muerta.

```
Run code x
C:\Users\Alumno\AppData\Local\Programs\Python\Python312\python.exe "E:\IFP\Sistemas\2aEv\Seguridad(script de escaneo)\escaneo\code.py"
Bienvenido al script de ping.
¿Qué desea hacer?
1. Ejecutar el código
2. Acceder a la documentación
3. Exportar resultados
4. Salir
Introduzca una opción: 1
Escaneando las IP's desde la : 192.168.128.1 hasta la 192.168.128.255
192.168.128.1 está viva!
192.168.128.2 está viva!
192.168.128.3 está viva!
192.168.128.4 está viva!
192.168.128.5 está viva!
192.168.128.6 está viva!
192.168.128.7 está viva!
192.168.128.8 está viva!
192.168.128.9 está viva!
192.168.128.10 está viva!
192.168.128.11 está viva!
192.168.128.12 está viva!
192.168.128.13 está viva!
192.168.128.14 está viva!
192.168.128.15 está viva!
192.168.128.16 está viva!
192.168.128.17 está viva!
192.168.128.18 está viva!
192.168.128.19 está viva!
192.168.128.20 está muerta!
192.168.128.21 está viva!
192.168.128.22 está muerta!
192.168.128.23 está muerta!
```

Imagen 15. Resultado del script de ping en Python

7.5)

La función de mostrar la documentación no tiene gran interés, ya que es un String en el que se explica qué hace el código, por lo que se pasa directamente a la función de exportación, en la cual, tras varios intentos de imprimir los resultados de la lista anterior sin éxito, se decide llamar al script de escaneo creado en el documento txt del [apartado 6](#).

```
usage
75 def exportar_resultados():
76     """
77     Exporta los resultados del ping a un documento en formato JSON.
78     resultados = []
79     """
80
81     print("Exportando datos, espere unos minutos...")
82     subprocess.call("escaneoExportar.cmd")
83     with open("sample.txt", "r") as f:
84         results = f.readlines()
85
86     results_json = [json.loads(line) for line in results]
87
88     with open("results.json", "w") as f:
89         json.dump(results_json, f)
90
91     print("¡Los resultados han sido exportados con éxito!")
92
93
```

Imagen 16. Exportar los resultados a JSON

Este proceso es bastante más largo de lo esperado, pero es funcional.

2.2. Módulo Network 2: Calculadora de direcciones IP en binario

2.2.1. Objetivo.

En este módulo el objetivo será crear un script para calcular las posibles combinaciones de una IP en función de su máscara.

2.2.2. Herramientas necesarias.

En este caso, viene dada una IP y una máscara para realizar el script:

- IP: 192.168.8.23.
- Máscara: 255.0.250.255.

2.2.3. Proceso de descarga de las herramientas necesarias.

No es necesaria la instalación de ningún programa para esta actividad.

2.2.4. Realización de la actividad.

1) *Realizar el cálculo de la red.*

Para obtener la red, se usa la IP y la máscara, realizando un AND entre cada octeto.

192 AND 250 (en binario sería igual a) = 11000000 AND 11111010 = **11000000**

168 AND 0 (en binario sería igual a) = 10101000 AND 00000000 = **00000000**

8 AND 250 (en binario sería igual a) = 00001000 AND 11111010 = **00001000**

23 AND 255 (en binario sería igual a) = 00010111 AND 11111111 = **00010111**

Por lo tanto, la red en binario sería la siguiente:

11000000.00000000.00001000.00010111

2) *Bits relevantes.*

El motivo de estudiar la red obtenida en binario es que, a la hora de obtener la red con la máscara, el valor del número de la máscara va a tener una importancia crucial para la obtención del script para la calculadora:

- Si el bit de la máscara **tiene como valor 1**, ese bit de la IP contra el que se compara es un **bit relevante**, y por lo tanto será **fijo**.

- Si el bit de la máscara **tiene como valor 0**, ese bit de la IP contra el que se compara **no es un bit relevante**, y por lo tanto será **variable**.

Conociendo esto, se puede saber qué bits de la red son relevantes y cuáles no: en los que sean relevantes, su valor se mantendrá fijo, y los no relevantes serán sustituidos por variables.

Máscara: 11111111.00000000.11111010.11111111

Red: 11000000.00000000.00001000.00010111

Sintaxis: 11000000.abcdefgh.00001i0j.00010111

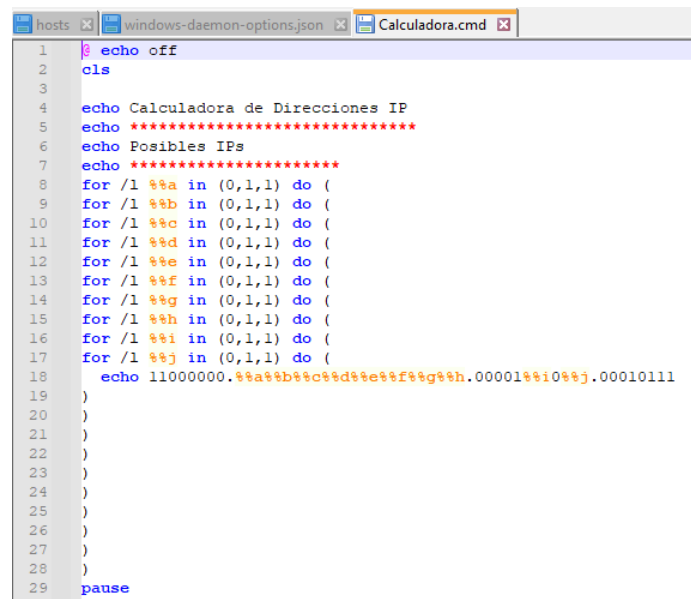
Por tanto, se puede observar que en los bits que como valor tenían un 1 en la máscara no varían, y los que tenían un 0 en la máscara son variables, que serán las que se usarán para determinar las distintas IPs de la red calculada.

3) Realización del script.

La creación del script se realiza en lenguaje batch, para ejecutarlo por la consola de Windows (CMD).

El script es realmente sencillo una vez obtenida la sintaxis, ya que su construcción se realizará creando bucles for anidados, que recorran cada una de las variables.

Estos bucles tendrán un rango establecido, que será entre cero y uno, y su aspecto sería el siguiente:



```

1  echo off
2  cls
3
4  echo Calculadora de Direcciones IP
5  echo *****
6  echo Posibles IPs
7  echo *****
8  for /l %a in (0,1,1) do (
9  for /l %b in (0,1,1) do (
10 for /l %c in (0,1,1) do (
11 for /l %d in (0,1,1) do (
12 for /l %e in (0,1,1) do (
13 for /l %f in (0,1,1) do (
14 for /l %g in (0,1,1) do (
15 for /l %h in (0,1,1) do (
16 for /l %i in (0,1,1) do (
17 for /l %j in (0,1,1) do (
18     echo 11000000.%a%b%c%d%e%f%g%h.00001%i0%j.00010111
19 )
20 )
21 )
22 )
23 )
24 )
25 )
26 )
27 )
28 )
29 pause

```

Imagen 17. Script en Notepad para la calculadora de IPs.

Ejecutando este script, se obtendrían ya todas las IPs de esta red, y a continuación se muestra un ejemplo del resultado del script ([Imagen 18](#)).

4) Implementación de un contador en el script.

Aunque con este script se obtienen todas las IPs de la red, habría que contar uno a uno los resultados de este para saber la cantidad de IPs que genera el comando.

Existe una fórmula para poder conocer la cantidad de IPs que se obtendrían del script sin ejecutarlo, que sería la siguiente:

$$2^{\text{número de variables del script}} - 2$$

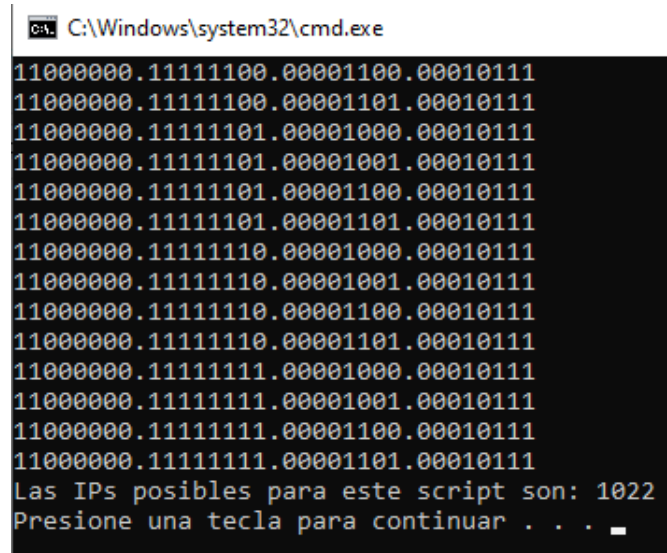
donde el número de variables del script son 10. El menos dos añadido al final indica que, normalmente, las IPs dedicadas al broadcast y al host no se cuentan como IPs validas.

En este caso, el resultado de este cálculo sería 1022 IPs posibles.

```
Seleccionar C:\Windows\system32\cmd.exe
Calculadora de Direcciones IP
*****
Posibles IPs
*****
11000000.00000000.00001000.00010111
11000000.00000000.00001001.00010111
11000000.00000000.00001100.00010111
11000000.00000000.00001101.00010111
11000000.00000001.00001000.00010111
11000000.00000001.00001001.00010111
11000000.00000001.00001100.00010111
11000000.00000001.00001101.00010111
11000000.00000010.00001000.00010111
11000000.00000010.00001001.00010111
11000000.00000010.00001100.00010111
11000000.00000010.00001101.00010111
11000000.00000011.00001000.00010111
11000000.00000011.00001001.00010111
11000000.00000011.00001100.00010111
11000000.00000011.00001101.00010111
11000000.00000100.00001000.00010111
11000000.00000100.00001001.00010111
11000000.00000100.00001100.00010111
11000000.00000100.00001101.00010111
11000000.00000101.00001000.00010111
11000000.00000101.00001001.00010111
11000000.00000101.00001100.00010111
11000000.00000101.00001101.00010111
11000000.00000110.00001000.00010111
11000000.00000110.00001001.00010111
11000000.00000110.00001100.00010111
11000000.00000110.00001101.00010111
11000000.00000111.00001000.00010111
11000000.00000111.00001001.00010111
11000000.00000111.00001100.00010111
11000000.00000111.00001101.00010111
11000000.00001000.00001000.00010111
11000000.00001000.00001001.00010111
11000000.00001000.00001100.00010111
11000000.00001000.00001101.00010111
11000000.00001001.00001000.00010111
11000000.00001001.00001001.00010111
11000000.00001001.00001100.00010111
11000000.00001001.00001101.00010111
11000000.00001010.00001000.00010111
11000000.00001010.00001001.00010111
11000000.00001010.00001100.00010111
11000000.00001010.00001101.00010111
```

Imagen 18. Script en CMD para la calculadora de IPs.

Para implementar este contador en el script, se inicializa un contador con valor -2, se añade un contador++ detrás del script a iterar para que aumente en uno el valor del contador cada vez que se obtenga una IP, y al final de todos los bucles anidados, se añade un “echo” para que muestre el valor del contador, como se muestra en las siguientes capturas.

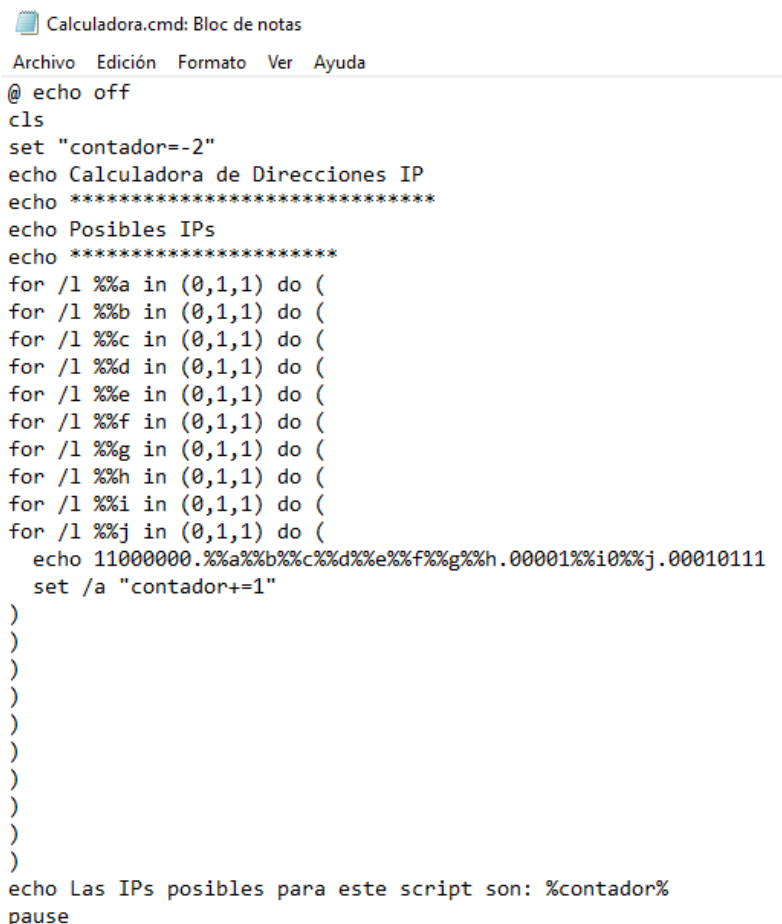


```

C:\Windows\system32\cmd.exe
11000000.11111100.00001100.00010111
11000000.11111100.00001101.00010111
11000000.11111101.00001000.00010111
11000000.11111101.00001001.00010111
11000000.11111101.00001100.00010111
11000000.11111101.00001101.00010111
11000000.11111110.00001000.00010111
11000000.11111110.00001001.00010111
11000000.11111110.00001100.00010111
11000000.11111110.00001101.00010111
11000000.11111111.00001000.00010111
11000000.11111111.00001001.00010111
11000000.11111111.00001100.00010111
11000000.11111111.00001101.00010111
Las IPs posibles para este script son: 1022
Presione una tecla para continuar . . .

```

Imagen 19. Resultado del contador del script en CMD.



```

Calculadora.cmd: Bloc de notas
Archivo Edición Formato Ver Ayuda
@ echo off
cls
set "contador=-2"
echo Calculadora de Direcciones IP
echo *****
echo Posibles IPs
echo *****
for /l %%a in (0,1,1) do (
for /l %%b in (0,1,1) do (
for /l %%c in (0,1,1) do (
for /l %%d in (0,1,1) do (
for /l %%e in (0,1,1) do (
for /l %%f in (0,1,1) do (
for /l %%g in (0,1,1) do (
for /l %%h in (0,1,1) do (
for /l %%i in (0,1,1) do (
for /l %%j in (0,1,1) do (
    echo 11000000.%%a%%b%%c%%d%%e%%f%%g%%h.00001%~i0%~j.00010111
    set /a "contador+=1"
)
)
)
)
)
)
)
)
)
)
)
echo Las IPs posibles para este script son: %contador%
pause

```

Imagen 20. Script en CMD para la calculadora de IPs con un contador implementado.

5) Exportación de datos a txt.

Por último, se necesita exportar los resultados de este script a un txt. Para ello, se realiza un segundo script en el cual se llame a este anterior, y se le añade un comando "> resultados.txt", que generará un documento con el resultado del script.

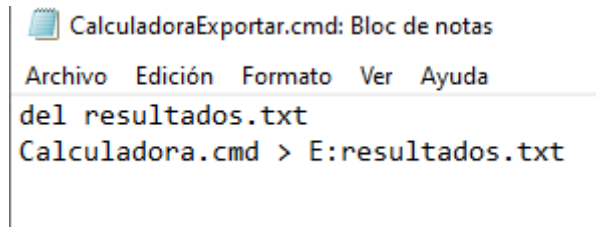


Imagen 21. Script para exportar los resultados de la calculadora.

2.3. Módulo Network 3: Sniffer de captura de tráfico de red.

2.3.1. Objetivo.

El objetivo de este módulo será utilizar una herramienta de sniffer de red para buscar vulnerabilidades en una página web dada durante un periodo de tiempo determinado.

2.3.2. Herramientas necesarias.

Para este módulo será necesaria la instalación de un programa conocido como Wireshark.

Esta herramienta no es más que un captador de tráfico de internet, con el cual se puede seguir el rastro a paquetes de datos, ver sus estadísticas y aplicar filtros para seleccionar los paquetes que se requieran a la hora de realizar una búsqueda.

A continuación, se hace un breve resumen para su instalación:

2.3.3. Proceso de descarga de las herramientas necesarias.

Para su instalación, es recomendable acudir a la web oficial del programa, que se adjunta a continuación:

<https://www.wireshark.org/download.html>

En dicha web, se selecciona la opción del instalador de Windows, el cual es el necesario en este caso.

Al descargarlo, se genera un archivo .exe, que se debe ejecutar para comenzar la instalación.

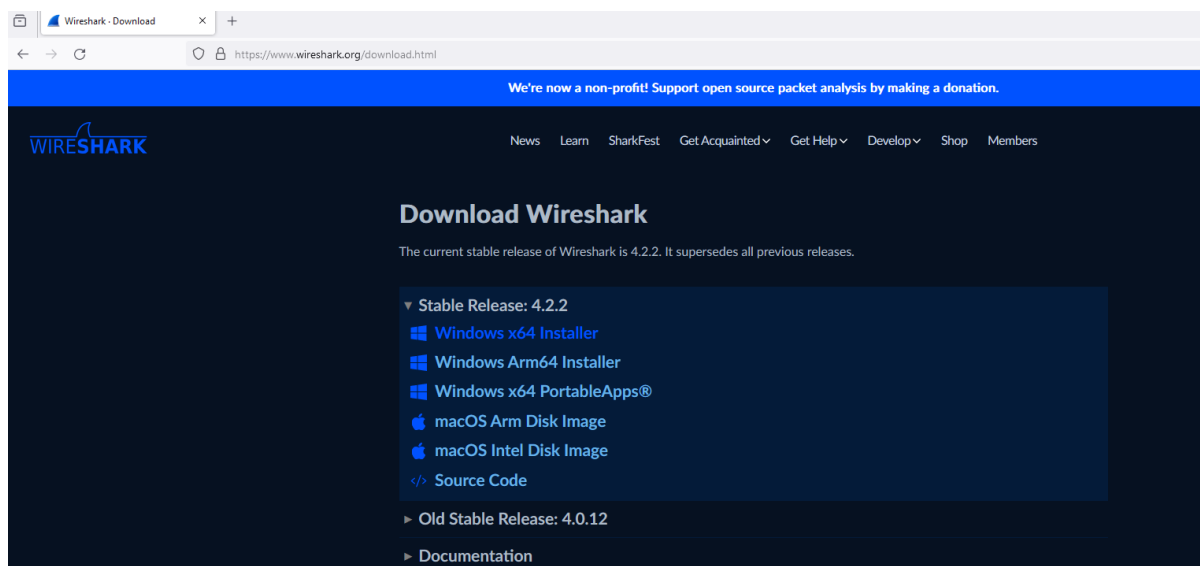


Imagen 22. Página oficial de WireShark

Tras abrir el .exe, la instalación es muy sencilla, ya que sólo habrá que ir aceptando todas las pantallas que vayan apareciendo.

De hecho, durante la instalación existe la opción de instalar el Nmap, una herramienta que, como se utilizará en el siguiente apartado y se comentará su instalación, no se va a instalar junto con este programa.

2.3.4. Realización de la actividad.

1) *Primeros pasos en la aplicación.*

Se comienza abriendo la aplicación del WireShark, en la cual aparecerán todos los medios posibles para aplicar el sniffer de red. De todos ellos, se debe seleccionar el medio que se esté utilizando para conectarse a la red, que en este caso concreto es el Ethernet.

Antes de hacer click y entrar a ver el tráfico de la red, es importante destacar que WireShark captura absolutamente todo el tráfico de red que esté activo en el ordenador, por lo que va a ser crucial el aislar la búsqueda necesaria en cada caso a través de los filtros que se explicarán más adelante, ya que, al guardar tanta información en memoria, la aplicación suele colgarse y es necesario cerrarla y reiniciar su proceso, lo que conlleva la pérdida de los paquetes que se querían analizar.

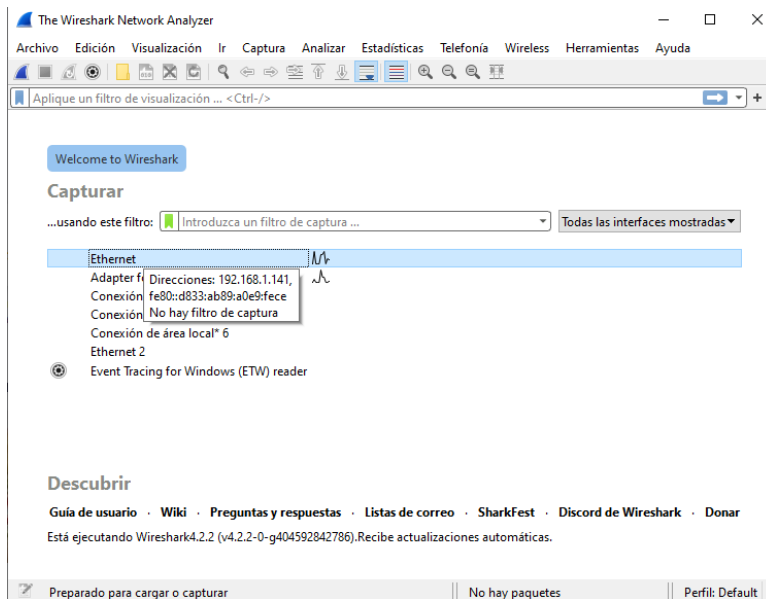


Imagen 23. Interfaz de inicio de WireShark

Una vez dentro del apartado ethernet, se accede a la interfaz principal de la herramienta, en la que, como hemos comentado antes, se observa una abrumadora cantidad de datos que aparecen cada pocos segundos en pantalla.

Esto representa **todo el tráfico de la red**, por lo que se capturan tanto paquetes de triple handshaking, protocolos tcp, icmp, https....

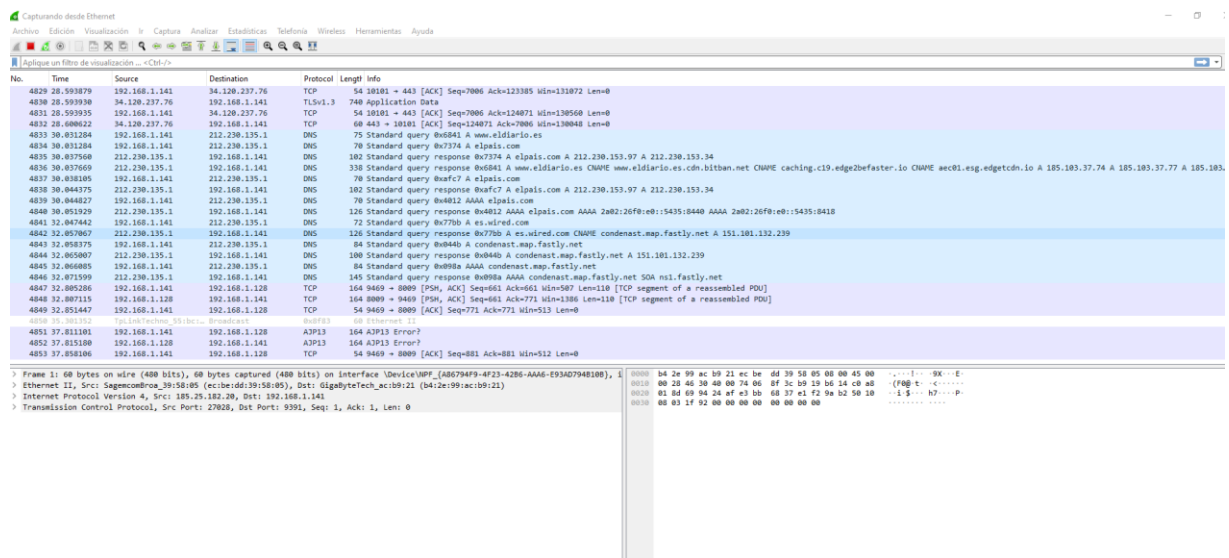


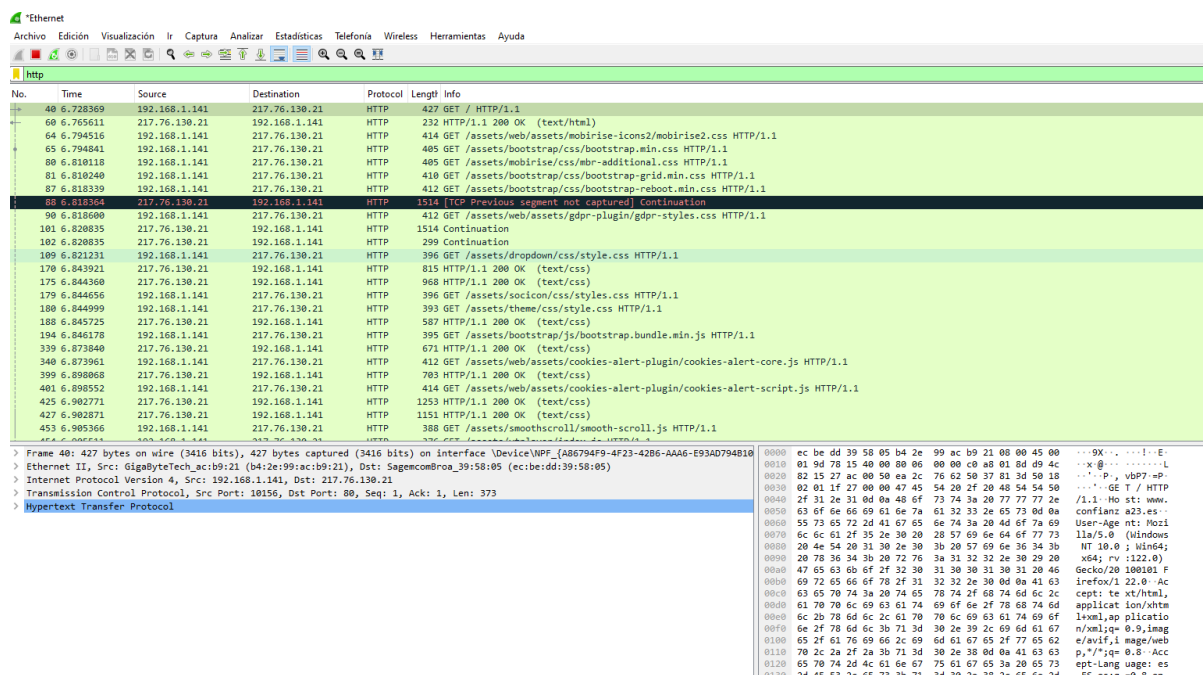
Imagen 24. Interfaz principal de WireShark con tráfico de todo tipo

En el caso de esta actividad, la página que se da al alumno para realizar la búsqueda en WireShark es la de confianza23, cuyo enlace se adjunta a continuación.

www.confianza23.es

2) Aplicación de filtros de búsqueda.

Para poder filtrar todo el tráfico que existe y centrar la búsqueda exclusivamente en la página dada, se va a aplicar el primer filtro en la barra superior, que especifica que es un área en la que se pueden establecer filtros. El filtro que se aplica es el de “http”, ya que la página tiene este protocolo de transferencia en su web, con lo que se acortaría un poco la búsqueda.



No.	Time	Source	Destination	Protocol	Length	Info
40	6.728369	192.168.1.141	217.76.130.21	HTTP	427	GET / HTTP/1.1
60	6.765611	217.76.130.21	192.168.1.141	HTTP	232	HTTP/1.1 200 OK (text/html)
64	6.794516	192.168.1.141	217.76.130.21	HTTP	414	GET /assets/web/assets/mobirise-icons2/mobirise2.css HTTP/1.1
65	6.794841	192.168.1.141	217.76.130.21	HTTP	405	GET /assets/bootstrap/css/bootstrap.min.css HTTP/1.1
80	6.818118	192.168.1.141	217.76.130.21	HTTP	405	GET /assets/mobirise/css/mbr-additional.css HTTP/1.1
81	6.818240	192.168.1.141	217.76.130.21	HTTP	410	GET /assets/bootstrap/css/bootstrap-grid.min.css HTTP/1.1
87	6.818339	192.168.1.141	217.76.130.21	HTTP	412	GET /assets/bootstrap/css/bootstrap-reboot.min.css HTTP/1.1
88	6.818364	217.76.130.21	192.168.1.141	HTTP	1514	[TCP Previous segment not captured] Continuation
90	6.818600	192.168.1.141	217.76.130.21	HTTP	412	GET /assets/web/assets/gdpr-plugin/gdpr-styles.css HTTP/1.1
101	6.828835	217.76.130.21	192.168.1.141	HTTP	1514	Continuation
102	6.828835	217.76.130.21	192.168.1.141	HTTP	299	Continuation
109	6.821231	192.168.1.141	217.76.130.21	HTTP	396	GET /assets/dropdown/css/style.css HTTP/1.1
170	6.843921	217.76.130.21	192.168.1.141	HTTP	815	HTTP/1.1 200 OK (text/css)
175	6.844360	217.76.130.21	192.168.1.141	HTTP	968	HTTP/1.1 200 OK (text/css)
179	6.844656	192.168.1.141	217.76.130.21	HTTP	396	GET /assets/socicon/css/styles.css HTTP/1.1
180	6.844999	192.168.1.141	217.76.130.21	HTTP	393	GET /assets/theme/css/style.css HTTP/1.1
188	6.845725	217.76.130.21	192.168.1.141	HTTP	587	HTTP/1.1 200 OK (text/css)
194	6.846178	192.168.1.141	217.76.130.21	HTTP	395	GET /assets/bootstrap/js/bootstrap.bundle.min.js HTTP/1.1
339	6.873840	217.76.130.21	192.168.1.141	HTTP	671	HTTP/1.1 200 OK (text/css)
340	6.873961	192.168.1.141	217.76.130.21	HTTP	412	GET /assets/web/assets/cookies-alert-plugin/cookies-alert-core.js HTTP/1.1
399	6.898068	217.76.130.21	192.168.1.141	HTTP	703	HTTP/1.1 200 OK (text/css)
401	6.898552	192.168.1.141	217.76.130.21	HTTP	414	GET /assets/web/assets/cookies-alert-plugin/cookies-alert-script.js HTTP/1.1
425	6.902771	217.76.130.21	192.168.1.141	HTTP	1253	HTTP/1.1 200 OK (text/css)
427	6.902871	217.76.130.21	192.168.1.141	HTTP	1151	HTTP/1.1 200 OK (text/css)
453	6.905366	192.168.1.141	217.76.130.21	HTTP	388	GET /assets/smoothscroll/smooth-scroll.js HTTP/1.1

Frame 40: 427 bytes on wire (3416 bits), 427 bytes captured (3416 bits) on interface 'DeviceNPF_{A86794F9-4F23-42B6-AA6E-E93AD794810}'
> Ethernet II, Src: GigabyteTech_ac:b9:21 (b4:2e:99:ac:b9:21), Dst: SagecomBroa_39:58:05 (ec:be:dd:39:58:05)
> Internet Protocol Version 4, Src: 192.168.1.141, Dst: 217.76.130.21
> Transmission Control Protocol, Src Port: 10156, Dst Port: 80, Seq: 1, Ack: 1, Len: 373
> Hypertext Transfer Protocol

0000 ec be dd 39 58 05 b4 2e 99 ac b9 21 08 00 45 00 ...X...:..E
0010 01 9d 78 15 40 00 00 06 00 00 c0 a0 01 8d 09 4c ...x@.....L
0020 82 15 27 ac 00 50 ea 3c 76 62 50 37 81 3d 50 18 ...P...vbP7-w
0030 02 01 1f 27 00 00 47 45 54 20 2f 20 48 54 50 ...GE T / HTTP
0040 2f 31 2e 31 0d 0a 48 6f 73 74 3a 20 77 77 72 e /1.1:Ho st: www.
0050 63 6f 6e 66 69 61 6e 7a 61 32 33 2e 65 73 00 0a confiam: a23.es-
0060 55 73 65 72 2d 41 67 65 6e 74 3a 20 4d 6f 7a 69 User-Age nt: Mozil
0070 6c 6e 61 2f 35 2e 30 20 28 57 69 6e 64 6f 77 73 lla/5.0 (Windows
0080 20 4e 54 20 31 30 2e 30 3b 20 57 69 6e 36 34 3b NT 10.0 ; Win64;
0090 20 78 36 34 3b 20 72 76 3a 31 32 32 2e 30 29 20 x64; rv:122.0)
00a0 47 65 63 0b 6f 2f 32 30 31 30 31 30 31 20 46 Gecko/20100101 F
00b0 69 72 65 06 6f 78 2f 31 32 32 2e 30 0d 0a 41 63 irefox/1.22.0 ;Ac
00c0 63 65 70 74 3a 20 74 65 78 74 2f 68 74 6d 6c 2c cept: te xt/html,
00d0 61 70 70 6c 69 63 61 74 69 6f 6e 2f 70 68 74 6d applicat ion/xhta
00e0 6c 2b 78 6d 6c 2c 61 70 70 6c 69 63 61 74 69 6f lxml,ap plicatio
00f0 6e 2f 78 6d 6c 3b 71 3d 30 2e 39 2c 69 6d 61 67 n/xml;q= 0.9,imag
0100 65 2f 61 76 69 66 2c 69 6d 61 67 65 2f 77 65 62 e/avi,f,1 mage/web
0110 70 2c 2a 2f 2a 3b 71 3d 30 2a 38 0d 0a 41 63 63 p,77;que 0.8;acc
0120 65 70 74 2d 4c 61 6e 67 75 61 67 65 3a 20 65 73 ept-Lang uage: es
0130 2d 4c 53 7c 65 73 3b 71 3d 3a 3b 78 7c 65 6a 2d -FC...a b 8 ..

Imagen 25. Tráfico con filtro http aplicado

3) Resolución de los apartados pedidos.

En el enunciado de la actividad se pide resolver las siguientes cuestiones:

- Dirección MAC de origen (resuelta)
- Dirección IP de origen
- Puerto de origen
- Dirección MAC de destino (no resuelta)
- Dirección IP de destino
- Puerto de destino (resuelto)
- Protocolo
- Campo de información

Además, se pide el archivo con los paquetes del triple handshaking, y que sólo alberguen los paquetes de datos de dicha web.

Los paquetes de esta web ya se han aislado filtrando por http, pero para sacar la información del triple handshaking, y ya de paso todo lo demás, se puede aplicar otro filtro de la siguiente manera:

- Se busca en cualquier paquete el **Stream Index**. Esta información, que aparece en el apartado de Transmission Control Protocol de la parte inferior

izquierda, es un número autogenerated por el WireShark que informa del paquete en el que se ha inicializado la petición GET, en la que el usuario pide al servidor acceder a la web.

- Se aplica click derecho sobre el Stream index, y se usa la opción de **aplicar como filtro seleccionado**. Esto va a acotar aun más la búsqueda, ya que sólo añadirá los paquetes que pertenecen a esa petición de acceso inicial a la web.

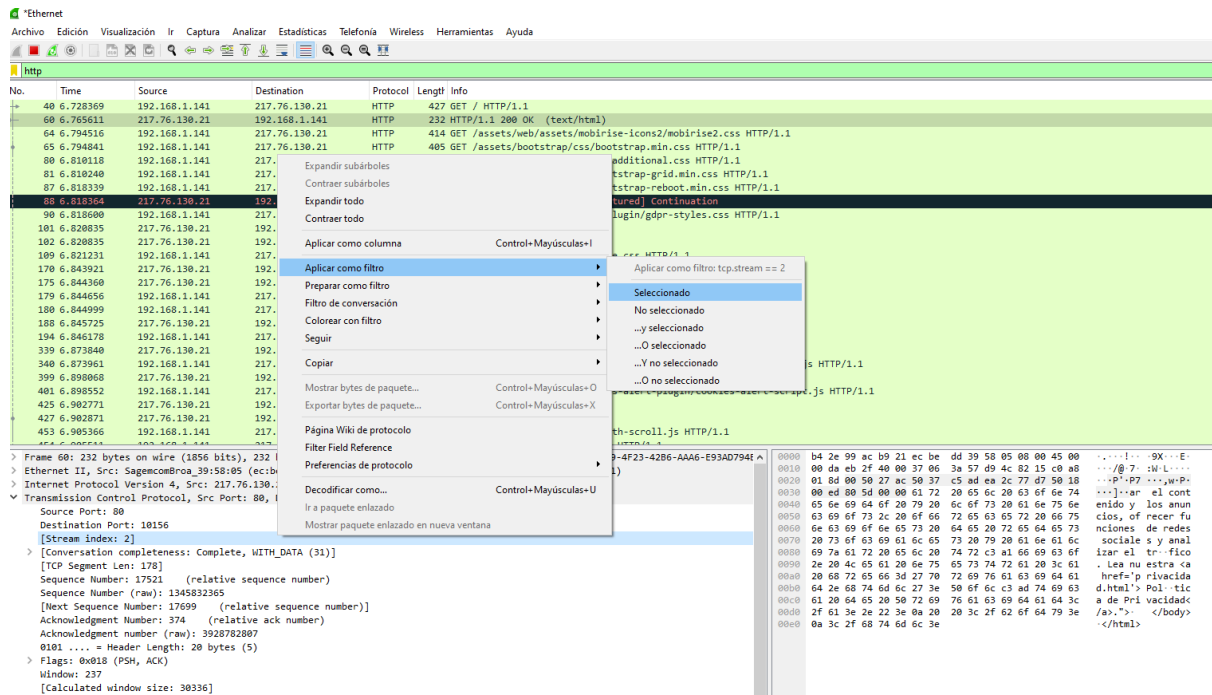


Imagen 26. Filtrado por Stream Index

- Si se fija la atención en los 4 primeros paquetes, se puede ver el protocolo de handshaking entre la IP del usuario y la web, en la que primero se manda un SYN desde la IP del usuario, después la web envía un SYN-ACK, y por último el usuario devuelve el ACK.

No.	Time	Source	Destination	Protocol	Length	Info
33	6.713677	192.168.1.141	217.76.130.21	TCP	66	10156 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
36	6.728227	217.76.130.21	192.168.1.141	TCP	66	80 → 10156 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM WS=128
38	6.728271	192.168.1.141	217.76.130.21	TCP	54	10156 → 80 [ACK] Seq=1 Ack=1 Win=131328 Len=0

Imagen 27. Protocolo handshaking

- Para poder conocer los paquetes que pertenecen exactamente a este protocolo, se pueden exportar como se muestra en la siguiente imagen.

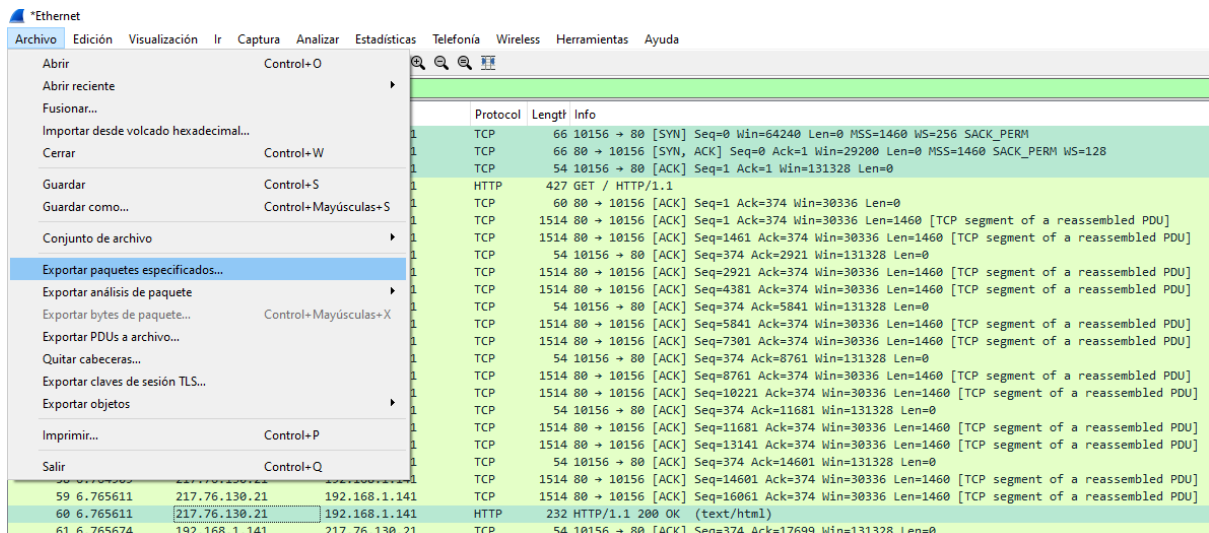


Imagen 28. Exportación de paquetes del handshaking

- Por último, si se quiere conocer la información de la MAC de origen, destino, puertos y demás, se puede analizar en el recuadro inferior izquierdo, en el que en las distintas pestañas se puede acceder a dichos datos.

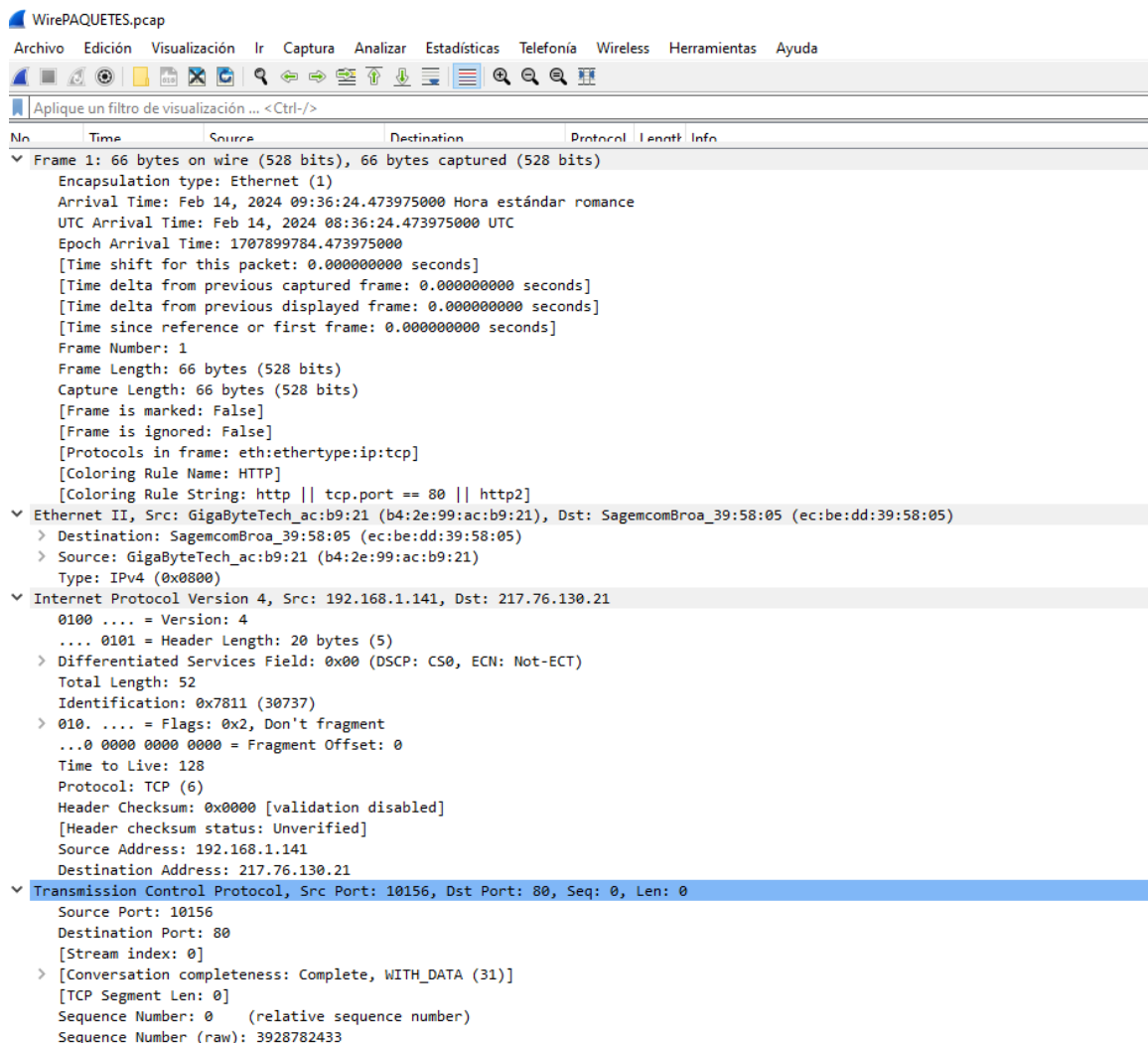


Imagen 29. Información de IP, MAC, puertos etc. pedidos en el enunciado

2.4. Módulo Network 4: Network Scanner, escaneador de red.

2.4.1. Objetivo.

El objetivo de este módulo será analizar los puertos activos de la IP del alumno, a través de un escaneador de red.

2.4.2. Herramientas necesarias.

Será necesaria la instalación del programa Zenmap. Este programa es una interfaz gráfica de la herramienta Nmap, que es un software de código abierto que permite realizar escaneo de redes y sus puertos, con el objetivo de conocer posibles brechas de seguridad en la misma.

A continuación, se hace un breve resumen para su instalación:

2.4.3. Proceso de descarga de las herramientas necesarias.

Para la instalación de cualquier programa es muy recomendable acudir al sitio web oficial, que en el caso del Nmap es el siguiente:

Download the Free Nmap Security Scanner for Linux/Mac/Windows

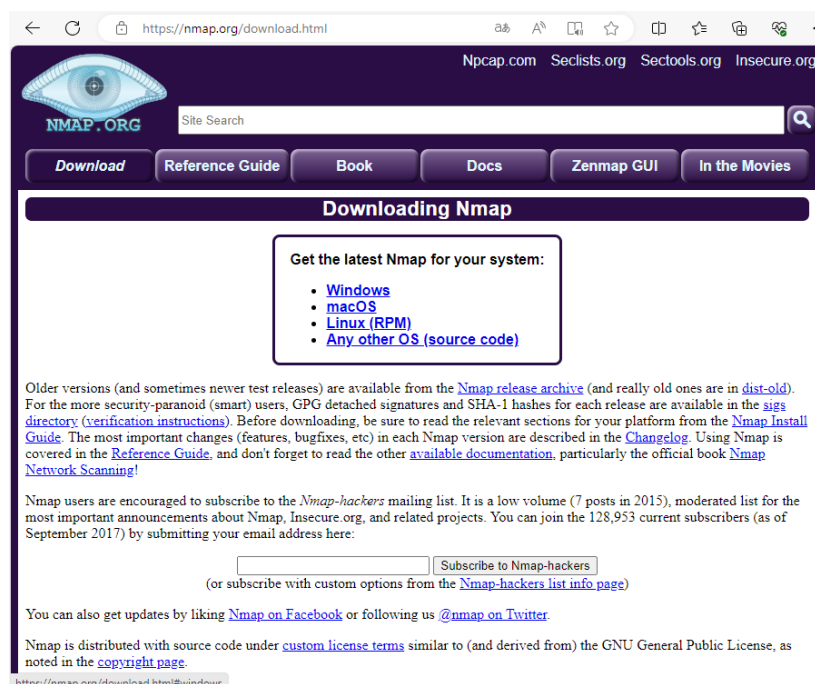
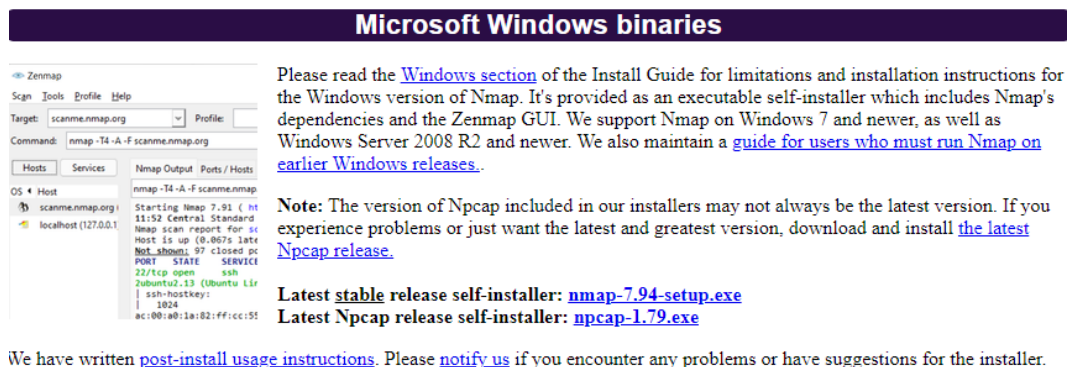


Imagen 30. Web oficial Nmap.

Una vez dentro, se selecciona el sistema operativo correspondiente (en el caso de este trabajo, se selecciona Windows), y se descarga la versión señalada como “Latest stable release self-installer”.



We have written [post-install usage instructions](#). Please [notify us](#) if you encounter any problems or have suggestions for the installer.

Imagen 31. Web oficial Nmap (instalador de Windows).

Después, solo será necesario ejecutar el archivo .exe descargado, y seguir las instrucciones para la instalación del programa, el cual es bastante sencillo. Una vez instalado, se puede proseguir con la práctica

2.4.4. Realización de la actividad.

1) Equipos visibles a nivel de IP:

Para conocer los equipos visibles a nivel de IP, existe un comando ya predeterminado en la aplicación de Zenmap, al cual podemos acceder de la siguiente manera:

- En la zona superior se pueden encontrar 3 zonas diferenciadas:
 - o **Target**, en la que se podrán aplicar rangos de IPs, puertos etc.
 - o **Profile**, donde aparecen los comandos rápidos.
 - o **Command**, en el que aparecen los comandos con los filtros aplicados.

Si se despliega la sección de Profile, se accede a los comandos ya creados por la aplicación.

Para este caso, el comando necesario sería el de **Ping scan**, que una vez seleccionado aparece de manera automática en el recuadro Command.

Por último, como solo se necesita realizar el ping de las IPs vivas de la red sobre la que se está realizando la actividad, se pondrá dicha red como un filtro en el recuadro de Target, quedando de la siguiente manera:

nmap -sn 192.168.128.0/24

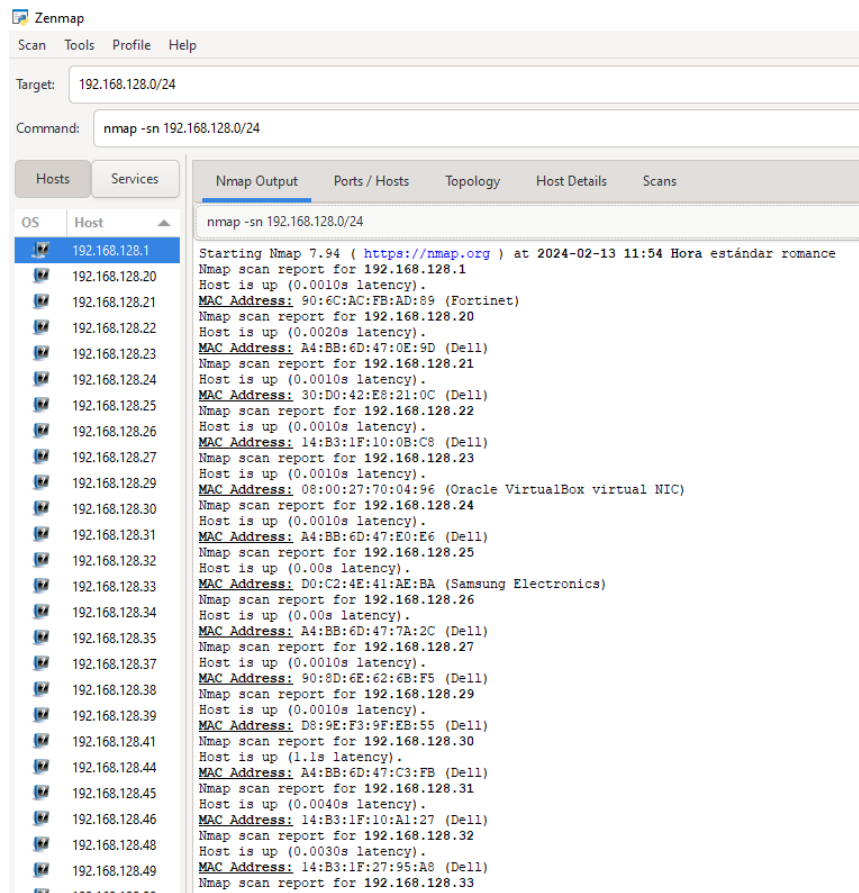


Imagen 32. Resultado del comando de escaneo de IPs en Zenmap

Para la exportación del resultado, se selecciona en la barra de herramientas superior de la aplicación el menú *scan*, *seleccionado Save Scan*, con lo que se obtiene un fichero XML con toda la información.

2) Listado de puertos TCP y UDP de los equipos de la red:

En este apartado se recurrirá de nuevo a un comando rápido de la barra Profile; en este caso se utilizará el comando llamado ***Intense scan, all TCP ports***.

Si sólo se ejecutara este comando, el proceso de escaneo se eternizaría, ya que este comando realiza un escaneo completo de toda la red. Para poder obtener unos resultados comprensibles en un tiempo razonable, se ejecuta el comando utilizando un filtro en el Target(que será la IP predeterminada del ordenador del alumno) quedando el siguiente comando:

`nmap -p 1-65535 -T4 -A -v 192.168.128.1`

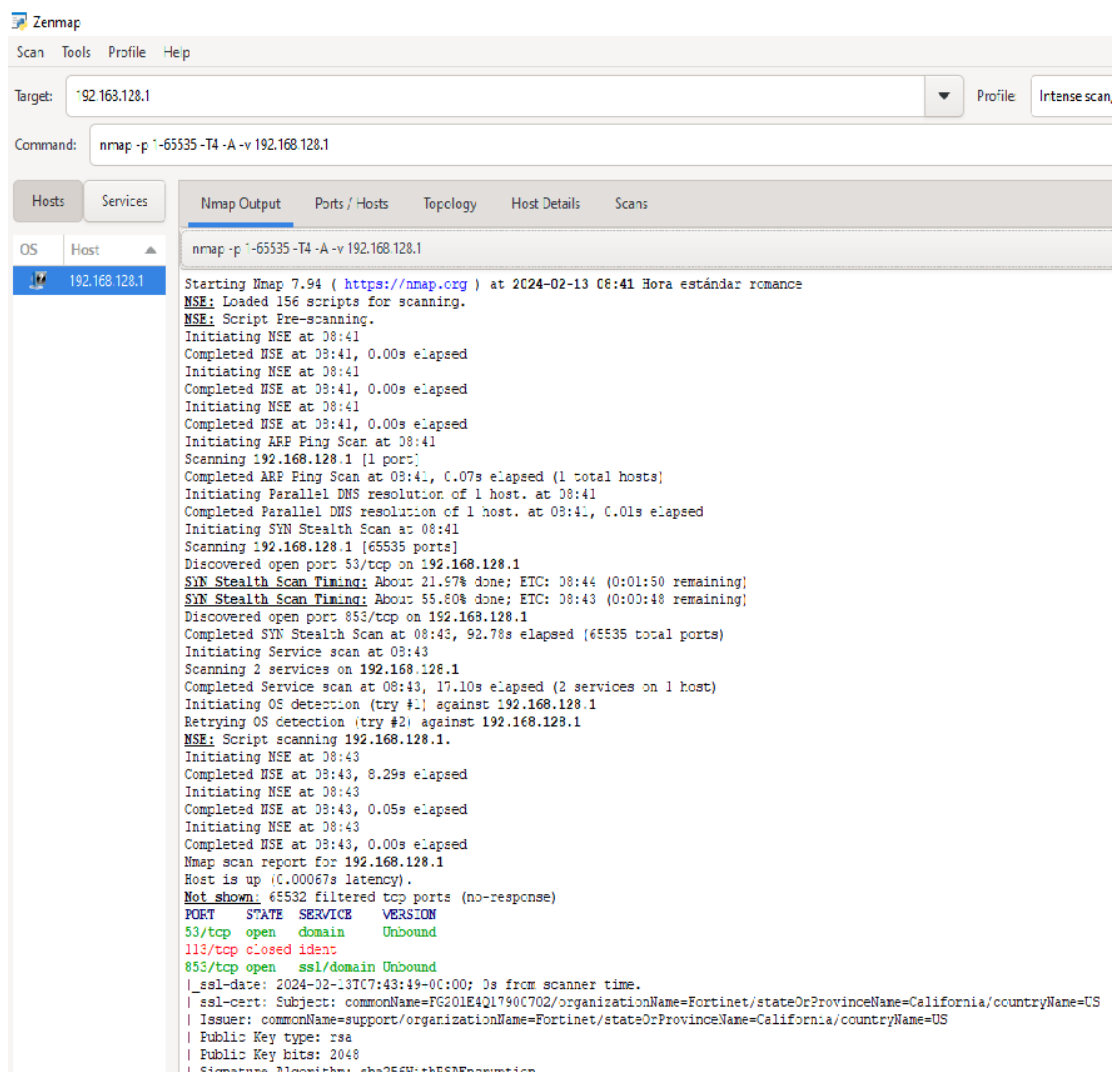


Imagen 33. Resultado de escaneo de comando de puertos TCP

Además, de este escaneo se pueden sacar cosas interesantes, ya que si se selecciona el apartado llamado **Ports/Host**, se pueden observar los puertos encontrados, tanto los abiertos como los cerrados.

Esto lleva a una pregunta importante: **¿qué significa que un puerto este abierto o cerrado?**

La respuesta a esta pregunta es sencilla: los puertos de un ordenador se pueden considerar como los puntos de acceso a una vivienda, ya que son lugares por los que se puede acceder a la misma. Al igual que durante una situación de robo, un ladrón puede entrar en la vivienda a través de esos puntos, en los ordenadores los puertos abiertos **pueden significar posibles puntos vulnerables** para acceder a la información del sistema del usuario, y corromperlo o obtener datos vulnerables.

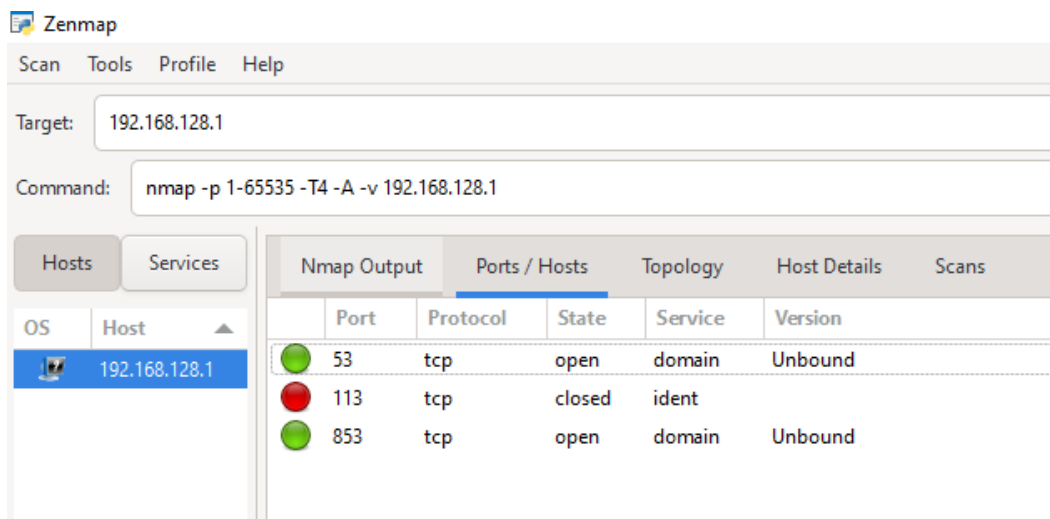


Imagen 34. Puertos encontrados tras el comando TCP

Estos puertos, de manera general, no se abren de manera consciente, sino que son abiertos por programas o aplicaciones específicas cuando es necesario acceder a ellos.

En este caso, se encuentran 2 puertos abiertos, y uno cerrado, los cuales se pueden estudiar a través de la tabla de puertos comunes de cualquier sistema, a la que se puede acceder a través del siguiente enlace:

<https://www.ionos.es/digitalguide/servidores/know-how/puertos-tcp-puertos-udp/>

- Puerto 53: Es utilizado por el servicio de DNS.
- Puerto 113: Ese puerto se utiliza para el servicio de identificación. Al acudir a la tabla mencionada anteriormente, se especifica que este puerto está en desuso, pero los programas lo utilizaban para conocer el usuario al que pertenece una conexión determinada.
- Puerto 853: No es uno de los puertos comunes.

Si en vez de realizar el comando para los TCP, se realizara para los UDP, se seleccionaría en la barra Profile la opción **Intense scan plus UDP**, y se aplicaría como filtro en el Target la IP de ruta predeterminada, siendo el comando el siguiente:

nmap -sS -sU -T4 -A -v 192.168.128.136

3) Servicio más probable en ejecución en cada puerto en base a IANA.

Antes de comentar el comando utilizado, se va a describir brevemente qué es la IANA.

Esta entidad es la encargada de asignar un número único a los distintos protocolos de internet para cada puerto, por lo que tiene un papel esencial en la organización del tráfico de datos de Internet.



Imagen 35. Funciones de la IANA. Fuente: <https://www.ionos.es/digitalguide/servidores/know-how/iana-que-es-y-cual-es-su-funcion/>

Si se acude a la página oficial de la IANA, se recoge información sobre qué rango de puertos están asignados a distintos protocolos, como se observa en la siguiente imagen.

```
Port numbers are assigned in various ways, based on three ranges: System Ports (0-1023), User Ports (1024-49151), and the Dynamic and/or Private Ports (49152-65535); the different uses of these ranges are described in [RFC6335]. According to Section 8.1.2 of [RFC6335], System Ports are assigned by the "IETF Review" or "IESG Approval" procedures described in [RFC8126]. User Ports are assigned by IANA using the "IETF Review" process, the "IESG Approval" process, or the "Expert Review" process, as per [RFC6335]. Dynamic Ports are not assigned.
```

Imagen 36. Asignación de puertos oficial de la IANA

Conocido esto, ya se puede dar paso a la ejecución del comando para ver los protocolos que pasarán por los puertos abiertos, según la IANA.

nmap -p 1-65535 192.168.128.136

Al ejecutar dicho comando, se obtienen los siguientes resultados:

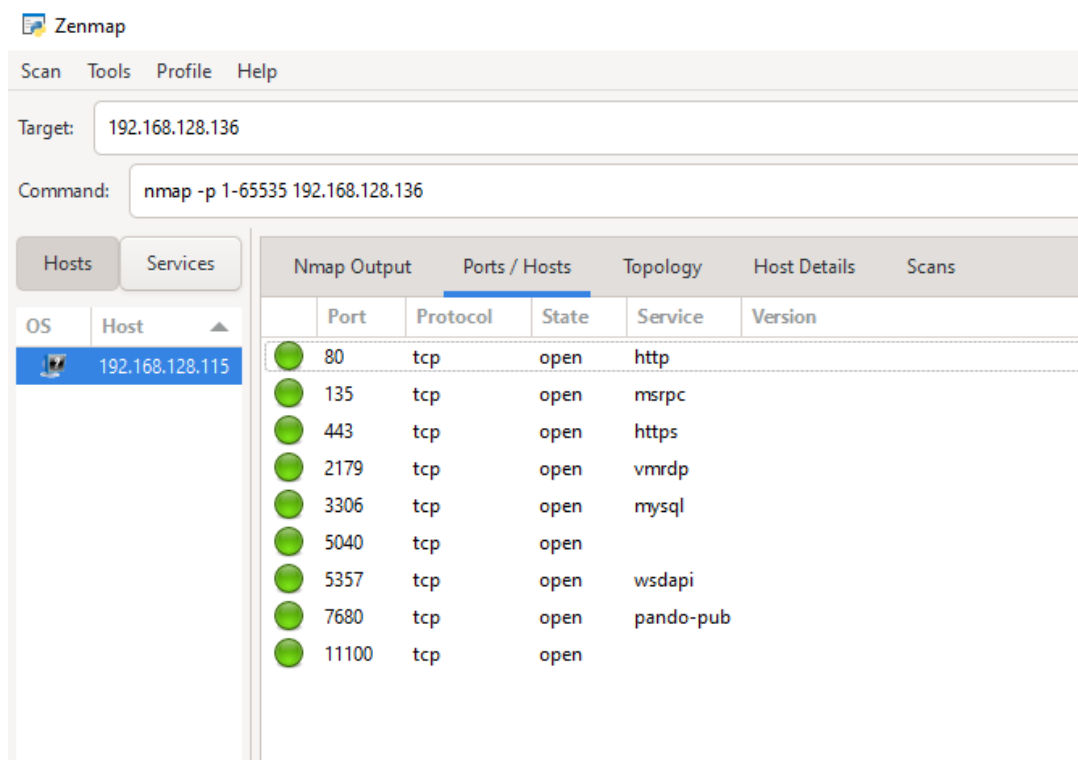


Imagen 37. Puertos probables por la IANA

3. Conclusiones del trabajo.

- Protocolo Ping:

- Herramienta fundamental para verificar la conectividad entre dos dispositivos en una red.
- Permite determinar la latencia y el tiempo de respuesta de un host.
- Útil para diagnosticar problemas de conexión y determinar la ruta de acceso a un destino.
- Limitado en cuanto a la información que proporciona sobre el estado de la red.

- Sniffers de Red:

- Permiten capturar y analizar el tráfico de red que circula por un segmento específico.
- Útiles para la monitorización de redes, la detección de intrusos y el análisis de protocolos.
- Pueden ser utilizados con fines maliciosos para interceptar información sensible.
- Es importante conocer las leyes y regulaciones relacionadas con el uso de sniffers de red.

- Escaneadores de Red:

- Permiten identificar y obtener información sobre los dispositivos conectados a una red.
- Útiles para la administración de redes, la identificación de vulnerabilidades y la evaluación de la seguridad.
- Algunos escáneres pueden ser utilizados con fines maliciosos para mapear redes y encontrar posibles puntos de ataque.
- Es importante utilizar los escáneres de red de forma responsable y con la autorización adecuada.

- **En general, el estudio de estas herramientas ha permitido:**

- Comprender mejor el funcionamiento de las redes TCP/IP.
- Identificar las diferentes técnicas utilizadas para la monitorización y el análisis de redes.
- Conocer los riesgos asociados al uso de estas herramientas.
- **Reflexionar sobre la importancia de la seguridad en las redes y la necesidad de utilizar estas herramientas de forma responsable.**

Este estudio ha servido como base para comprender las diferentes técnicas y herramientas utilizadas para la gestión y la seguridad de las redes. Se espera que los resultados de este trabajo puedan ser utilizados para mejorar la seguridad y la eficiencia de las redes informáticas personales en un futuro.

- **Limitaciones:**

- Este estudio se ha centrado en una descripción general de las herramientas mencionadas.
- No se ha profundizado en el uso específico de cada herramienta.
- No se han abordado las diferentes opciones de configuración y personalización de estas herramientas.

4. Tabla resumen de imágenes.

Imagen 1. Página oficial de PyCharm	4
Imagen 2. Ipconfig del equipo del alumno.....	5
Imagen 3. Ping a la IP de la ruta predeterminada	6
Imagen 4. Bucle for para realizar 3 veces un ping	7
Imagen 5. Ejecución del bucle for para 3 pings.....	7
Imagen 6. Bucle for para las Ips acabadas entre 1 y 3	7
Imagen 7. Ejecución del bucle for para ping a Ips acabadas entre 1 y 3	8
Imagen 8. Bucle for para todas las Ips entre la 1 y la 255.....	8
Imagen 9. Bucle for anidado para todo el rango de equipos de la red.	9
Imagen 10. Find TTL del script de escaneo	9
Imagen 11. Imports de Python	10
Imagen 12. Clase bcolors obtenida en StackOverflow	11
Imagen 13. Implementación de menú con Google Bard	12
Imagen 14. Script de ping en Python	12
Imagen 15. Resultado del script de ping en Python	13
Imagen 16. Exportar los resultados a JSON	13
Imagen 17. Script en Notepad para la calculadora de IPs.	15
Imagen 18. Script en CMD para la calculadora de IPs.....	16
Imagen 19. Resultado del contador del script en CMD.	17
Imagen 20. Script en CMD para la calculadora de IPs con un contador implementado.....	17
Imagen 21. Script para exportar los resultados de la calculadora.	18
Imagen 22. Página oficial de WireShark	19
Imagen 23. Interfaz de inicio de WireShark.....	20
Imagen 24. Interfaz principal del WireShark con tráfico de todo tipo	20
Imagen 25. Tráfico con filtro http aplicado.....	21
Imagen 26. Filtrado por Stream Index	22
Imagen 27. Protocolo handshaking	22
Imagen 28. Exportación de paquetes del handshaking.....	23
Imagen 29. Información de IP, MAC, puertos etc. pedidos en el enunciado	23
Imagen 30. Web oficial Nmap.	24
Imagen 31. Web oficial Nmap (instalador de Windows).	25
Imagen 32. Resultado del comando de escaneo de IPs en Zenmap	26
Imagen 33. Resultado de escaneo de comando de puertos TCP	27
Imagen 34. Puertos encontrados tras el comando TCP.....	28
Imagen 35. Funciones de la IANA. Fuente: https://www.ionos.es/digitalguide/servidores/know-how/iana-que-es-y-cual-es-su-funcion/	29
Imagen 36. Asignación de puertos oficial de la IANA.....	29
Imagen 37. Puertos probables por la IANA	30