

# S11L5

## Indice

1. Spiegazione Esercizio
2. Utilizzo di Windows PowerShell
3. Analisi Traffico HTTP e HTTPS
4. Analisi del Comando Nmap
5. Analisi di un Attacco Sql Injection
6. Conclusioni

## 1 - Spiegazione Esercizio

L'obiettivo della giornata è quello di affrontare un'analisi approfondita di vari laboratori tecnici, focalizzandosi su argomenti di rilevante importanza per la sicurezza informatica e l'amministrazione dei sistemi. In particolare, verranno esplorate diverse aree tematiche, a partire dall'utilizzo di PowerShell in ambiente Windows, una tecnologia versatile e potente per l'automazione e la gestione dei sistemi operativi Microsoft.

Parallelamente, sarà dedicato ampio spazio allo studio del traffico di rete, con un focus specifico sui protocolli HTTP e HTTPS. Questa parte del laboratorio mira a fornire una comprensione profonda delle differenze strutturali tra le due modalità di comunicazione, evidenziando le implicazioni in termini di sicurezza e prestazioni. Verranno analizzati gli aspetti tecnici legati alla crittografia, all'uso dei certificati digitali e alle potenziali vulnerabilità che possono essere sfruttate durante lo scambio di dati su rete.

La sessione successiva sarà incentrata su due tipologie di attacchi informatici, con l'obiettivo di comprendere non solo le tecniche utilizzate, ma anche le metodologie di rilevamento e mitigazione. Il primo caso di studio riguarderà l'utilizzo di Nmap, uno strumento fondamentale per la scansione e l'analisi delle reti, esplorandone le capacità di rilevare dispositivi attivi, servizi esposti e configurazioni potenzialmente vulnerabili. L'attenzione si focalizzerà sia sugli aspetti tecnici dell'attacco sia sulle contromisure che possono essere adottate per proteggere le infrastrutture.

Infine, verrà analizzato un attacco mirato a un database MySQL. Questo laboratorio metterà in luce le modalità di compromissione di un sistema di gestione dei database, illustrando come un attaccante possa sfruttare vulnerabilità comuni per ottenere accesso non autorizzato ai dati. Particolare enfasi sarà posta sull'importanza di configurazioni sicure, gestione degli account e implementazione di misure preventive per ridurre al minimo i rischi.

Questa esplorazione si propone non solo di fornire competenze tecniche pratiche, ma anche di sviluppare un approccio critico e proattivo nei confronti della sicurezza e dell'amministrazione dei sistemi.

## 2 - Utilizzo di Windows PowerShell

### Che cosa è Windows PowerShell? Che differenze ci sono con la prompt dei comandi?

Windows PowerShell è uno strumento di automazione e gestione dei sistemi operativi Windows, progettato per facilitare il controllo e la configurazione di sistemi locali e remoti attraverso un ambiente basato su comandi e script. È basato su .NET Framework e successivamente su .NET Core (con PowerShell Core e poi PowerShell 7), il che gli consente di sfruttare le potenzialità di un linguaggio di scripting avanzato e versatile.

A differenza della tradizionale Prompt dei Comandi (cmd), che è un'interfaccia testuale basata su MS-DOS e utilizza comandi semplici per eseguire operazioni di base sul sistema, PowerShell offre funzionalità molto più potenti e flessibili. Alcune differenze principali sono le seguenti:

#### 1. Lingua di Scripting Avanzata

PowerShell non è solo un interprete di comandi, ma un vero e proprio linguaggio di scripting basato su .NET. Ciò consente di scrivere script complessi con strutture di controllo come cicli, condizioni e funzioni. Inoltre, supporta oggetti, il che lo rende particolarmente utile per manipolare dati in formati complessi.

Al contrario, la Prompt dei Comandi offre solo script batch limitati, che non hanno la stessa ricchezza di costrutti logici e strumenti.

#### 2. Uso degli Oggetti

Una delle caratteristiche distintive di PowerShell è il suo approccio basato su oggetti. I comandi (o cmdlet) di PowerShell non restituiscono solo testo, ma oggetti completi che possono essere manipolati direttamente. Questo significa che è possibile accedere a proprietà e metodi di un oggetto senza dover fare parsing manuale.

La Prompt dei Comandi, invece, produce solo output testuali, il che richiede manipolazioni manuali per estrarre informazioni utili.

#### 3. Cmdlet vs Comandi Tradizionali

PowerShell utilizza una serie di cmdlet predefiniti, che sono piccoli comandi con una struttura uniforme del tipo **Verbo-Sostantivo** (ad esempio, **Get-Process**, **Set-Service**, **Remove-Item**). Questa uniformità semplifica l'apprendimento e la comprensione dei comandi.

La Prompt dei Comandi utilizza comandi tradizionali come **dir**, **copy**, o **del**, che spesso hanno sintassi incoerenti e limitate.

#### 4. Integrazione con Windows e Funzionalità Avanzate

PowerShell è profondamente integrato con Windows e consente di interagire facilmente con il registro di sistema, i servizi, il WMI (Windows Management Instrumentation), e altre API. Può essere utilizzato per gestire processi, configurare server, e persino amministrare ambienti cloud.

La Prompt dei Comandi, invece, è limitata alle funzioni di base del sistema operativo e non offre integrazioni dirette con questi strumenti avanzati.

## 5. Espandibilità e Moduli

PowerShell supporta l'aggiunta di moduli, che sono pacchetti contenenti cmdlet, funzioni e script per gestire specifiche tecnologie o applicazioni (ad esempio, Azure, Active Directory, SQL Server). Inoltre, è possibile creare cmdlet personalizzati per esigenze particolari.

La Prompt dei Comandi non ha un sistema equivalente di estendibilità.

## 6. Cross-Platform

Le versioni moderne di PowerShell (a partire da PowerShell Core) sono multi-piattaforma e possono essere eseguite su Windows, macOS e Linux, offrendo un ambiente unificato per la gestione di sistemi eterogenei.

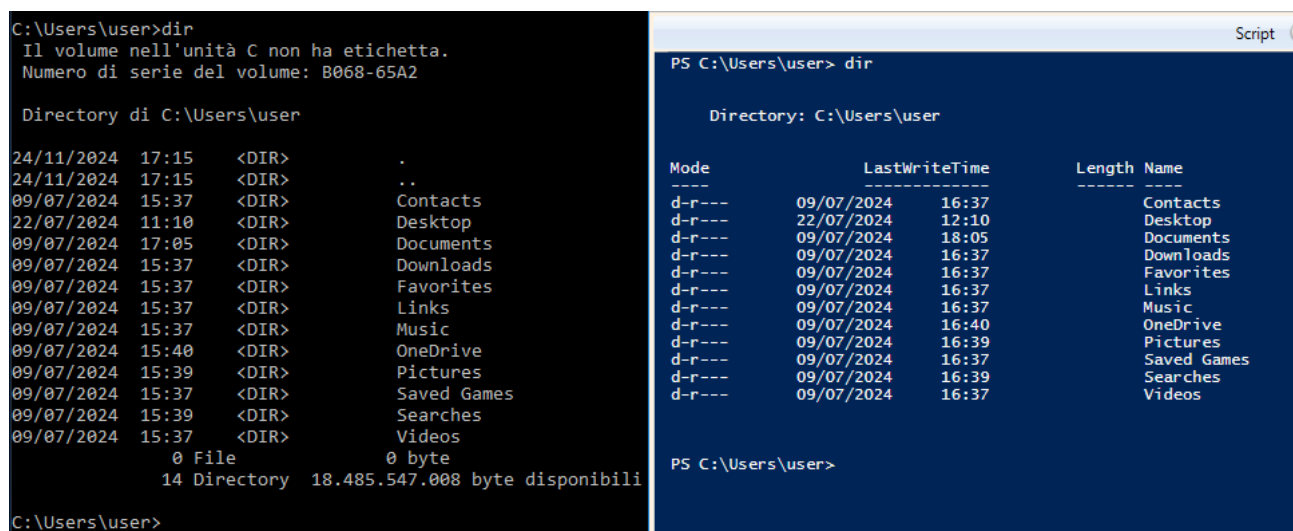
La Prompt dei Comandi è esclusiva per Windows.

In sintesi, PowerShell è uno strumento molto più potente e versatile rispetto alla Prompt dei Comandi, concepito per automatizzare e gestire ambienti complessi in modo efficiente. La Prompt dei Comandi rimane utile per operazioni semplici o per motivi di retrocompatibilità, ma PowerShell rappresenta l'evoluzione moderna e robusta per gli amministratori di sistema e gli sviluppatori.

## Parte Pratica

Come primo passo bisogna aprire e utilizzare contemporaneamente sia la Prompt dei Comandi che la PowerShell. Dopodiché sarà essenziale provare ogni comando in entrambi i terminali in modo da notare le differenze tra i due.

Partiamo con un comando base come **'dir'**.



The image shows two side-by-side terminal windows. The left window is the Windows Command Prompt (CMD) and the right window is PowerShell (PS).

**Left Window (CMD):**

```
C:\Users\user>dir
Il volume nell'unità C non ha etichetta.
Numero di serie del volume: B068-65A2

Directory di C:\Users\user

24/11/2024  17:15    <DIR>        .
24/11/2024  17:15    <DIR>        ..
09/07/2024  15:37    <DIR>        Contacts
22/07/2024  11:10    <DIR>        Desktop
09/07/2024  17:05    <DIR>        Documents
09/07/2024  15:37    <DIR>        Downloads
09/07/2024  15:37    <DIR>        Favorites
09/07/2024  15:37    <DIR>        Links
09/07/2024  15:37    <DIR>        Music
09/07/2024  15:40    <DIR>        OneDrive
09/07/2024  15:39    <DIR>        Pictures
09/07/2024  15:37    <DIR>        Saved Games
09/07/2024  15:39    <DIR>        Searches
09/07/2024  15:37    <DIR>        Videos
               0 File             0 byte
              14 Directory  18.485.547.008 byte disponibili

C:\Users\user>
```

**Right Window (PowerShell):**

```
PS C:\Users\user> dir

Directory: C:\Users\user

Mode                LastWriteTime         Length Name
----                -
d-r---             09/07/2024   16:37             Contacts
d-r---             22/07/2024   12:10             Desktop
d-r---             09/07/2024   18:05             Documents
d-r---             09/07/2024   16:37             Downloads
d-r---             09/07/2024   16:37             Favorites
d-r---             09/07/2024   16:37             Links
d-r---             09/07/2024   16:37             Music
d-r---             09/07/2024   16:40             OneDrive
d-r---             09/07/2024   16:39             Pictures
d-r---             09/07/2024   16:37             Saved Games
d-r---             09/07/2024   16:39             Searches
d-r---             09/07/2024   16:37             Videos

PS C:\Users\user>
```

Entrambi gli ambienti Windows forniscono un elenco di sotto directory e file, insieme a informazioni associate come tipo, dimensione del file, data e ora dell'ultima modifica. In PowerShell, vengono mostrati anche gli attributi/modalità.

Ora proviamo con comandi diversi come ‘**ipconfig**’, ‘**ping**’ e ‘**cd**’.

Schermata che mostra il comando ‘**ipconfig**’:

```
09/07/2024 17:05 <DIR> Documents
09/07/2024 15:37 <DIR> Downloads
09/07/2024 15:37 <DIR> Favorites
09/07/2024 15:37 <DIR> Links
09/07/2024 15:37 <DIR> Music
09/07/2024 15:40 <DIR> OneDrive
09/07/2024 15:39 <DIR> Pictures
09/07/2024 15:37 <DIR> Saved Games
09/07/2024 15:39 <DIR> Searches
09/07/2024 15:37 <DIR> Videos
0 File 0 byte
14 Directory 18.485.547.088 byte disponibili

C:\Users\user>ipconfig

Configurazione IP di Windows

Scheda Ethernet Ethernet:

    Suffisso DNS specifico per connessione:
    Indirizzo IPv6 . . . . . : 2a01:e11:4004:9c30:959:2601:f044:b9ec01:f044:b9ec
    Indirizzo IPv6 temporaneo. . . . . : 2a01:e11:4004:9c30:f52e:b3b:5b29:8ac3b:5b29:8ac
    Indirizzo IPv6 locale rispetto al collegamento . : fe80::959:2601:f044:b9ec%4
    Indirizzo IPv4. . . . . : 192.168.1.138
    Subnet mask . . . . . : 255.255.255.0
    Gateway predefinito . . . . . : fe80::3a07:16ff:fe1a:ad28%4
    192.168.1.254

Scheda Tunnel isatap.{92D61F82-1D19-45C9-B7CF-2E5AF2D63627}:

    Stato supporto. . . . . : Supporto disconnesso
    Suffisso DNS specifico per connessione:

Scheda Tunnel Teredo Tunneling Pseudo-Interface:

    Suffisso DNS specifico per connessione:
    Indirizzo IPv6 . . . . . : 2001:0:2851:782c:47f:fa8:aec7:3c1a
    Indirizzo IPv6 locale rispetto al collegamento . : fe80::47f:fa8:aec7:3c1a%5
    Gateway predefinito . . . . . :

C:\Users\user>
```

```
PS C:\Users\user> ipconfig

Configurazione IP di Windows

Scheda Ethernet Ethernet:

    Suffisso DNS specifico per connessione:
    Indirizzo IPv6 . . . . . : 2a01:e11:4004:9c30:959:2601:f044:b9ec01:f044:b9ec
    Indirizzo IPv6 temporaneo. . . . . : 2a01:e11:4004:9c30:f52e:b3b:5b29:8ac3b:5b29:8ac
    Indirizzo IPv6 locale rispetto al collegamento . : fe80::959:2601:f044:b9ec%4
    Indirizzo IPv4. . . . . : 192.168.1.138
    Subnet mask . . . . . : 255.255.255.0
    Gateway predefinito . . . . . : fe80::3a07:16ff:fe1a:ad28%4
    192.168.1.254

Scheda Tunnel isatap.{92D61F82-1D19-45C9-B7CF-2E5AF2D63627}:

    Stato supporto. . . . . : Supporto disconnesso
    Suffisso DNS specifico per connessione:

Scheda Tunnel Teredo Tunneling Pseudo-Interface:

    Suffisso DNS specifico per connessione:
    Indirizzo IPv6 . . . . . : 2001:0:2851:782c:47f:fa8:aec7:3c1a
    Indirizzo IPv6 locale rispetto al collegamento . : fe80::47f:fa8:aec7:3c1a%5
    Gateway predefinito . . . . . :

PS C:\Users\user>
```

Schermata che mostra il comando ‘**ping**’:

```
C:\Users\user>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=17ms TTL=120
Risposta da 8.8.8.8: byte=32 durata=18ms TTL=120
Risposta da 8.8.8.8: byte=32 durata=18ms TTL=120
Risposta da 8.8.8.8: byte=32 durata=18ms TTL=120
Risposta da 8.8.8.8: byte=32 durata=18ms TTL=120

Statistiche Ping per 8.8.8.8:
    Pacchetti: Trasmessi = 4, Ricevuti = 4,
    Persi = 0 (0% persi),
    Tempo approssimativo percorsi andata/ritorno in millisecondi:
    Minimo = 17ms, Massimo = 18ms, Medio = 17ms
```

```
PS C:\Users\user> 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=17ms TTL=120
Risposta da 8.8.8.8: byte=32 durata=17ms TTL=120
Risposta da 8.8.8.8: byte=32 durata=18ms TTL=120
Risposta da 8.8.8.8: byte=32 durata=17ms TTL=120

Statistiche Ping per 8.8.8.8:
    Pacchetti: Trasmessi = 4, Ricevuti = 4,
    Persi = 0 (0% persi),
    Tempo approssimativo percorsi andata/ritorno in millisecondi:
    Minimo = 17ms, Massimo = 18ms, Medio = 17ms

PS C:\Users\user> |
```

Schermata che mostra il comando ‘**cd**’:

```
C:\Users\user>cd
C:\Users\user

C:\Users\user>
```

```
PS C:\Users\user> cd
PS C:\Users\user> |
```

Come si può notare dalle schermate la risposta su entrambi i terminali è pressoché la stessa.

Ora andiamo ad esplorare il comando **Netstat** usando unicamente la **PowerShell**.

Il primo comando che andrò ad effettuare sarà '**Netstat -h**' che mi permetterà di visualizzare la totalità dei comandi netcat a disposizione.

```
PS C:\Users\user> netstat -h
netstat :
In riga:1 car:1
+ netstat -h
+ ~~~~~
+ CategoryInfo          : NotSpecified: (String) [], RemoteException
+ FullyQualifiedErrorId : NativeCommandError

Visualizza statistiche relative ai protocolli e alle
connessioni di rete TCP/IP correnti.
NETSTAT [-a] [-b] [-e] [-f] [-n] [-o] [-p proto] [-r] [-s] [-x] [-t] [interval]
-a Visualizza tutte le connessioni e le porte di ascolto.
-b Visualizza il file eseguibile utilizzato per la creazione
  di ogni connessione o porta di ascolto. Alcuni file
  eseguibili conosciuti includono pi- componenti indipendenti.
  In tali casi viene visualizzata la sequenza dei componenti
  utilizzati per la creazione della connessione o porta di
  ascolto e il nome del file eseguibile viene visualizzato
  in fondo, tra parentesi quadre ([]). Nella parte superiore
  S indicato il componente chiamato e cos via, fino al
  raggiungimento di TCP/IP. Se si utilizza questa opzione,
  l'esecuzione del comando pu- richiedere molto tempo e
  riuscir- solo se si dispone di autorizzazioni sufficienti.
-e Visualizza le statistiche Ethernet. Pu- essere utilizzata
  insieme all'opzione -s.
-f Visualizza i nomi di dominio completi (FQDN, Fully Qualified
  Domain Name) per gli indirizzi esterni.
-n Visualizza indirizzi e numeri di porta in forma numerica.
-o Visualizza l'ID del processo proprietario associato a ogni
  connessione.
-p proto Visualizza le connessioni relative al protocollo specificato
  da "proto", che pu- essere TCP, UDP, TCPv6 o UDPv6.
  Se utilizzato insieme all'opzione -s per le statistiche per
  protocollo, "proto" pu- essere: IP, IPv6, ICMP, ICMPv6, TCP,
  TCPv6, UDP o UDPv6.
-q Visualizza tutte le connessioni, le porte di ascolto e le porte
  TCP non di ascolto associate. Le porte non di ascolto associate
  possono essere associate o meno a una connessione attiva.
-r Visualizza la tabella di routing.
-s Visualizza le statistiche per protocollo. Per impostazione
  predefinita, vengono visualizzate le statistiche per IP,
  IPv6, ICMP, ICMPv6, TCP, TCPv6, UDP e UDPv6. Per specificare
  un sottoinsieme dei valori predefiniti, S possibile
  utilizzare l'opzione -p.
-t Visualizza lo stato di offload della connessione corrente.
-x Visualizza le connessioni, i listener e gli endpoint
```

Dopo aver visualizzato tutti i comandi, andrò ad utilizzare il comando '**netstat -r**'. Che mi permetterà di visualizzare la tabella di Routing.

```
PS C:\Users\user> netstat -r

=====
Elenco interfacce
4...08 00 27 b8 e0 bd .....Intel(R) PRO/1000 MT Desktop Adapter
1.....Software Loopback Interface 1
6...00 00 00 00 00 00 00 e0 Microsoft ISATAP Adapter
5...00 00 00 00 00 00 00 e0 Microsoft Teredo Tunneling Adapter
=====

IPv4 Tabella route
=====
Route attive:
Indirizzo rete      Mask      Gateway    Interfaccia  Metrica
0.0.0.0             0.0.0.0    192.168.1.254  192.168.1.138    10
127.0.0.0           255.0.0.0    On-link      127.0.0.1      306
127.0.0.1           255.255.255.255  On-link      127.0.0.1      306
127.255.255.255     255.255.255.255  On-link      127.0.0.1      306
192.168.1.0         255.255.255.0   On-link      192.168.1.138   266
192.168.1.138       255.255.255.255  On-link      192.168.1.138   266
192.168.1.255       255.255.255.255  On-link      192.168.1.138   266
224.0.0.0           240.0.0.0    On-link      127.0.0.1      306
224.0.0.0           240.0.0.0    On-link      192.168.1.138   266
255.255.255.255     255.255.255.255  On-link      127.0.0.1      306
255.255.255.255     255.255.255.255  On-link      192.168.1.138   266
=====
```

In questa schermata possiamo vedere ad esempio l'ip gateway '**192.168.1.254**'

Per procedere, è necessario aprire PowerShell con privilegi di amministratore. Una volta aperta, è possibile utilizzare il comando **netstat -abno**.

Questo comando serve a monitorare le connessioni di rete attive e offre una panoramica dettagliata delle porte utilizzate e dei processi associati.

```
PS C:\Windows\system32> netstat -abno

Connessioni attive

Proto Indirizzo locale          Indirizzo esterno    Stato      PID
-----
TCP    0.0.0.0:7                    0.0.0.0:0           LISTENING  2064
[tcpvcs.exe]
TCP    0.0.0.0:9                    0.0.0.0:0           LISTENING  2064
[tcpvcs.exe]
TCP    0.0.0.0:13                   0.0.0.0:0           LISTENING  2064
[tcpvcs.exe]
TCP    0.0.0.0:17                   0.0.0.0:0           LISTENING  2064
[tcpvcs.exe]
TCP    0.0.0.0:19                   0.0.0.0:0           LISTENING  2064
[tcpvcs.exe]
TCP    0.0.0.0:80                   0.0.0.0:0           LISTENING  4
Impossibile ottenere informazioni sulla propriet...
TCP    0.0.0.0:135                  0.0.0.0:0           LISTENING  780
RpcSs
[svchost.exe]
TCP    0.0.0.0:445                  0.0.0.0:0           LISTENING  4
Impossibile ottenere informazioni sulla propriet...
TCP    0.0.0.0:1801                 0.0.0.0:0           LISTENING  1532
[mqsvc.exe]
TCP    0.0.0.0:2103                 0.0.0.0:0           LISTENING  1532
[mqsvc.exe]
TCP    0.0.0.0:2105                 0.0.0.0:0           LISTENING  1532
[mqsvc.exe]
TCP    0.0.0.0:2107                 0.0.0.0:0           LISTENING  1532
[mqsvc.exe]
TCP    0.0.0.0:2869                 0.0.0.0:0           LISTENING  4
Impossibile ottenere informazioni sulla propriet...
TCP    0.0.0.0:3389                 0.0.0.0:0           LISTENING  912
TermService
[svchost.exe]
TCP    0.0.0.0:5432                 0.0.0.0:0           LISTENING  2728
[postgres.exe]
TCP    0.0.0.0:8009                 0.0.0.0:0           LISTENING  2268
[tomcat7.exe]
TCP    0.0.0.0:8080                 0.0.0.0:0           LISTENING  2268
[tomcat7.exe]
TCP    0.0.0.0:8443                 0.0.0.0:0           LISTENING  4
Impossibile ottenere informazioni sulla propriet...
TCP    0.0.0.0:49408                0.0.0.0:0           LISTENING  492
Impossibile ottenere informazioni sulla propriet...
TCP    0.0.0.0:49409                0.0.0.0:0           LISTENING  108
EventLog
[svchost.exe]
TCP    0.0.0.0:49410                0.0.0.0:0           LISTENING  920
Schedule
```

Per confrontarle possiamo aprire la gestione attività e confrontare i PID a disposizione.

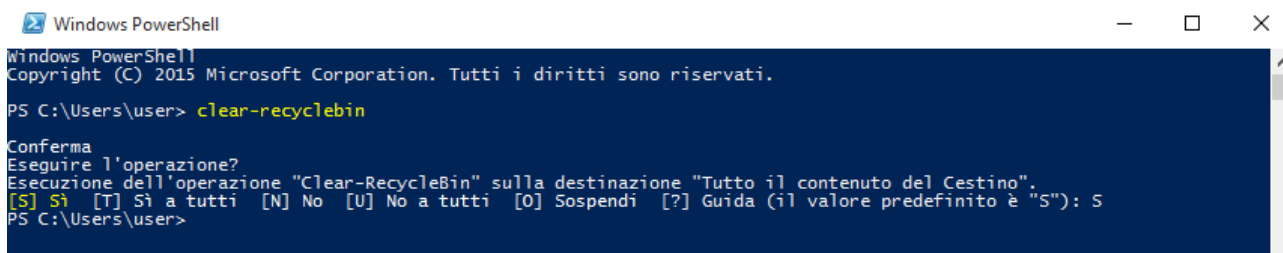
Gestione attività						
File Opzioni Visualizza						
Processi Prestazioni Cronologia applicazioni Avvio Utenti Dettagli Servizi						
Nome	PID	Stato	Nome utente	CPU	Memoria (...)	Descrizione
csrss.exe	424	In esecuzione	SYSTEM	00	648 K	Processo runtime client server
wininit.exe	492	In esecuzione	SYSTEM	00	448 K	Applicazione di avvio di Windows
csrss.exe	508	In esecuzione	SYSTEM	00	572 K	Processo runtime client server
winlogon.exe	576	In esecuzione	SYSTEM	00	736 K	Applicazione Accesso a Windows
services.exe	620	In esecuzione	SYSTEM	00	2,048 K	App Servizi e Controller
lsass.exe	640	In esecuzione	SYSTEM	00	3,720 K	Local Security Authority Process
svchost.exe	724	In esecuzione	SYSTEM	00	3,196 K	Processo host per servizi di Windows
svchost.exe	780	In esecuzione	SERVIZIO DI RETE	00	2,384 K	Processo host per servizi di Windows
svchost.exe	912	In esecuzione	SERVIZIO DI RETE	00	5,464 K	Processo host per servizi di Windows
svchost.exe	920	In esecuzione	SYSTEM	00	12,884 K	Processo host per servizi di Windows
dwm.exe	972	In esecuzione	DWM-1	00	22,396 K	Gestione finestre desktop
svchost.exe	1028	In esecuzione	SYSTEM	00	39,652 K	Processo host per servizi di Windows
svchost.exe	1040	In esecuzione	SYSTEM	00	3,844 K	Processo host per servizi di Windows
svchost.exe	1124	In esecuzione	SERVIZIO LOCALE	00	4,464 K	Processo host per servizi di Windows
svchost.exe	1136	In esecuzione	SERVIZIO LOCALE	00	400 K	Processo host per servizi di Windows

Il PID 756 è associato al processo **svchost.exe**. L'utente di questo processo è **NETWORK SERVICE** e sta utilizzando **4132K di memoria**.

Infine, PowerShell può essere utilizzata per gestire diverse parti del sistema, inclusi elementi specifici come il Cestino. Ad esempio, tramite il comando **Clear-RecycleBin** è possibile svuotare completamente il Cestino, eliminando tutti i file in esso contenuti. Questo comando offre un metodo rapido ed efficace per liberare spazio e mantenere il sistema ordinato, automatizzando un'operazione che altrimenti richiederebbe l'intervento manuale tramite l'interfaccia grafica.



Schermata che mostra il comando 'Clear-recyclebin':



```
Windows PowerShell
Copyright (C) 2015 Microsoft Corporation. Tutti i diritti sono riservati.

PS C:\Users\user> 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
PS C:\Users\user>
```

## 3 - Analisi Traffico Http e Https

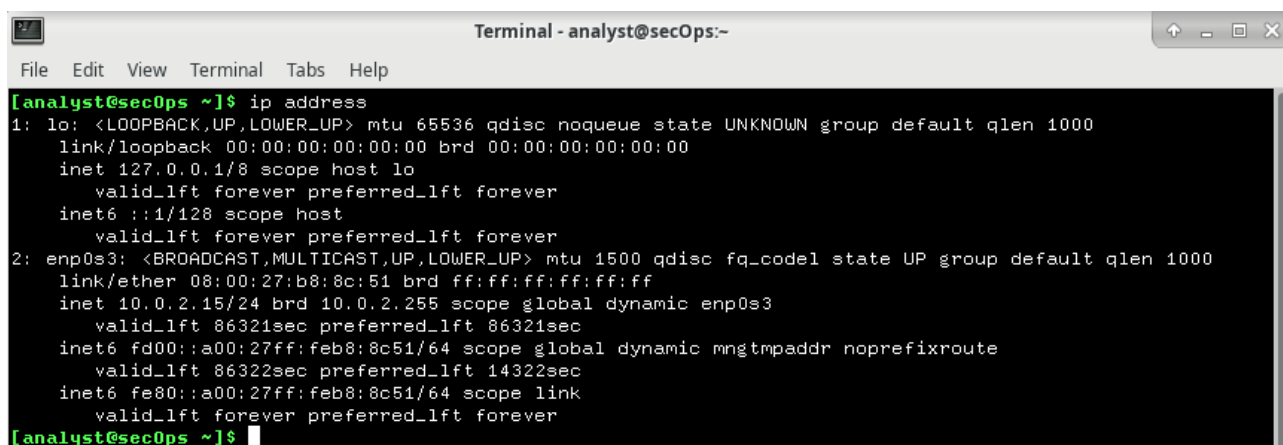
### Come si effettua un'analisi del traffico Http e Https?

Per analizzare il traffico di rete utilizzerò un software chiamato **Wireshark**, uno strumento avanzato che consente di visualizzare i pacchetti in entrata e in uscita dalla rete. Grazie alla sua interfaccia grafica e ai filtri avanzati, è possibile isolare facilmente specifici tipi di traffico, come i pacchetti **HTTP** o **HTTPS**, per concentrarsi su protocolli o comunicazioni particolari.

Tuttavia, per ottenere una cattura precisa e mirata di un determinato momento del traffico di rete, farò ricorso al comando **tcpdump**. Questo strumento a riga di comando è ideale per acquisire pacchetti in tempo reale e salvare le catture in un file da analizzare successivamente con Wireshark, consentendo così un'analisi più dettagliata e flessibile.

### Parte Pratica

Il primo passo consiste nell'utilizzare il comando **ip address**, che permette di ottenere informazioni fondamentali sulla configurazione della rete della macchina. Questo comando fornisce dettagli come gli indirizzi IP assegnati alle interfacce di rete, i gateway, la subnet mask e lo stato delle connessioni. Queste informazioni sono essenziali per comprendere la configurazione di rete attuale e per eseguire analisi o troubleshooting in modo più efficace.



```
Terminal - analyst@secOps:~
File Edit View Terminal Tabs Help

[analyst@secOps ~]$ ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:b8:8c:51 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic enp0s3
        valid_lft 86321sec preferred_lft 86321sec
    inet6 fd00::a00:27ff:feb8:8c51/64 scope global dynamic mngtmpaddr noprefixroute
        valid_lft 86322sec preferred_lft 14322sec
    inet6 fe80::a00:27ff:feb8:8c51/64 scope link
        valid_lft forever preferred_lft forever

[analyst@secOps ~]$
```

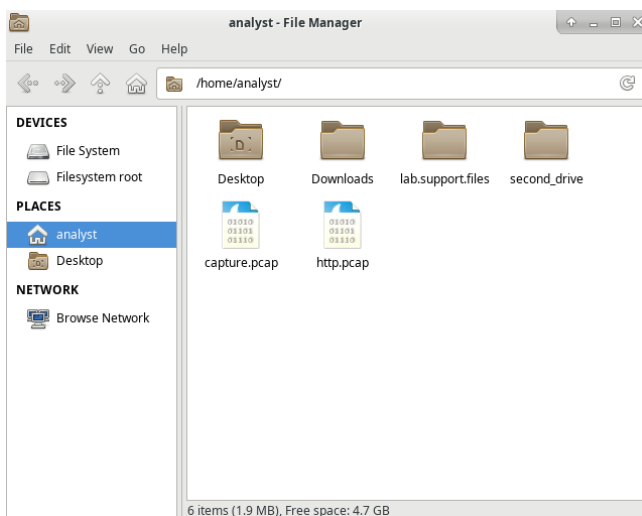
Grazie alle informazioni ottenute con il comando precedente, possiamo ora eseguire il comando **sudo tcpdump -i enp0s3 -s 0 -w httpdump.pcap**. Questo comando permette di avviare una cattura del traffico di rete in tempo reale utilizzando **tcpdump**, specificando alcune opzioni importanti. L'opzione **-i enp0s3** indica a **tcpdump** di monitorare l'interfaccia di rete **enp0s3**, che rappresenta una delle interfacce

di rete della macchina (questa variabile può variare in base alla configurazione del sistema). L'opzione **-s 0** è utilizzata per catturare l'intero pacchetto, evitando che tcpdump tronchi i pacchetti più grandi e fornendo una visione completa dei dati in transito. Infine, con **-w httpdump.pcap** si specifica il nome del file in cui verranno salvati i pacchetti catturati, in questo caso **httpdump.pcap**, che potrà essere successivamente analizzato con strumenti come Wireshark. In sintesi, il comando permette di registrare tutto il traffico che passa attraverso l'interfaccia di rete specificata e di salvare i dati in un file, pronto per un'analisi approfondita.

```
[analyst@secOps ~]$ sudo tcpdump -i enp0s3 -s 0 -w http.pcap
[sudo] password for analyst:
tcpdump: listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
^C1921 packets captured
1921 packets received by filter
0 packets dropped by kernel
[analyst@secOps ~]$
```

Dopo avviato la cattura, bisogna andare sul seguente sito: <http://www.altoromutual.com/login.jsp> e accedere alla piattaforma utilizzando le credenziali 'Admin' sia per l'username che per la password. É importante subito dopo chiudere il browser e chiudere la cattura con il comando 'ctrl + c'.

Dopo aver fatto ciò troveremo questo file: 'http.pcap'

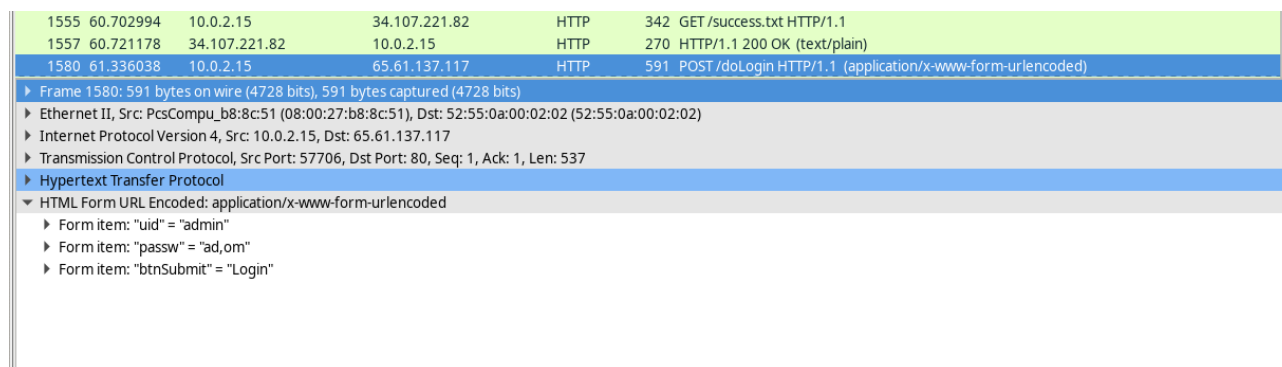


Facendo doppio click sul file, ci verrà mostrata la cattura direttamente sul software wireshark.

Con **Wireshark** aperto, procederemo ad analizzare in dettaglio la richiesta **POST** inviata per trasmettere l'username e la password del nostro account, utilizzati per l'autenticazione. In particolare, esamineremo il traffico di rete per identificare il momento in cui le credenziali vengono inviate dal client al server, osservando le informazioni contenute nel pacchetto, come l'header della richiesta e il corpo della stessa, dove i dati sensibili potrebbero essere trasmessi. Questo tipo di analisi ci permette di comprendere meglio come avviene la comunicazione durante il processo di login e di valutare eventuali vulnerabilità o problematiche legate alla sicurezza del traffico.



Schermata che mostra la richiesta 'Post':



La mancata crittografia del protocollo **HTTP** è evidente nella schermata, poiché Wireshark mostra il pacchetto in chiaro, con le credenziali (username e password) visibili all'interno del corpo della richiesta. Poiché il traffico HTTP non è protetto da crittografia, le informazioni sensibili vengono trasmesse in formato leggibile, rendendo possibile il loro accesso a chiunque abbia la capacità di intercettare il traffico di rete. Questa mancanza di protezione evidenzia un rischio significativo per la sicurezza, poiché le credenziali possono essere facilmente catturate e sfruttate da malintenzionati.

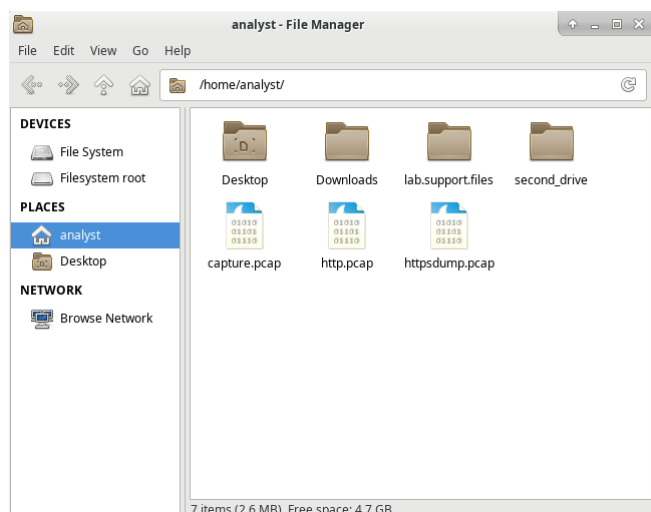
Ora andremo ad effettuare un'ulteriore cattura del traffico ma in questo caso con il protocollo **HTTPS**.

In questo caso il comando per effettuare la cattura sarà: **sudo tcpdump -i enp0s3 -s 0 -w httpsdump.pcap**

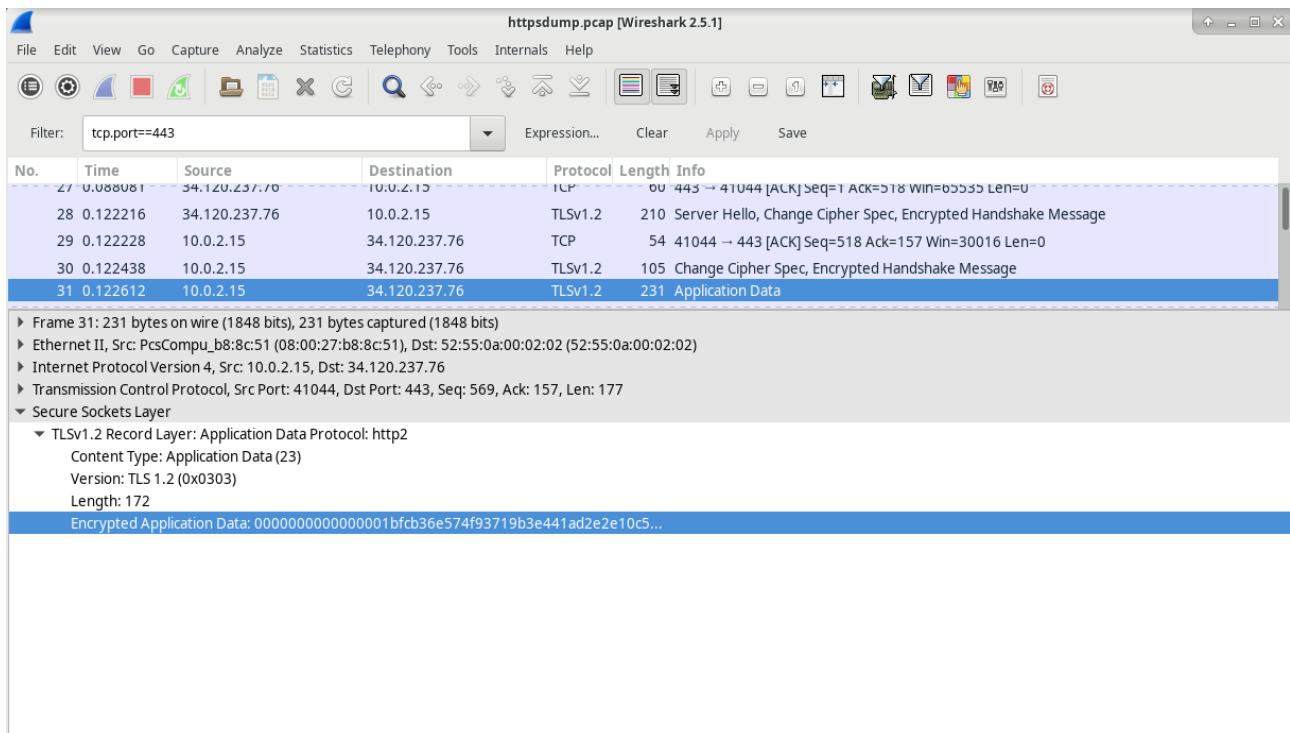
```
[analyst@secOps ~]$ sudo tcpdump -i enp0s3 -s 0 -w httpsdump.pcap
tcpdump: listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
^C920 packets captured
920 packets received by filter
0 packets dropped by kernel
```

Successivamente, ci conatteremo al sito [www.netacad.com](http://www.netacad.com) e tenteremo di effettuare il login. Anche se useremo un'email non valida, ciò non rappresenterà un problema, poiché l'obiettivo principale è analizzare le richieste **HTTPS**. In particolare, ci concentreremo sul monitoraggio del traffico crittografato generato durante il tentativo di autenticazione, osservando come le informazioni vengano scambiate tra il nostro browser e il server, al fine di comprendere meglio il flusso di comunicazione sicuro.

Il file di cattura lo troveremo nella stessa directory precedente:



Anche in questo caso, il file catturato verrà aperto con **Wireshark**, e procederemo ad analizzare i pacchetti che utilizzano il protocollo **TLSv1.2**, che è il protocollo impiegato per la crittografia dei dati. Durante questa analisi, esamineremo i dettagli del pacchetto, ma poiché i dati sono criptati, non riusciremo a visualizzare alcuna informazione sensibile in chiaro. Questo ci permette di osservare la struttura e le negoziazioni del protocollo sicuro, senza poter accedere ai contenuti effettivi, che sono protetti dalla crittografia.



## 4 - Analisi comando Nmap

### Che cosa è il comando Nmap?

**Nmap** (Network Mapper) è uno strumento di scansione di rete open source ampiamente utilizzato per mappare e analizzare reti di computer. È uno dei tool più potenti e versatili nel campo della sicurezza informatica, utilizzato per scoprire dispositivi attivi in una rete, rilevare le porte aperte, determinare i servizi che i dispositivi stanno eseguendo e identificare eventuali vulnerabilità. La sua principale funzione è quella di eseguire una scansione delle reti per determinare quali sistemi sono connessi, quali porte sono aperte e quali servizi sono in esecuzione su quelle porte.

Questo è utile per gli amministratori di rete, che utilizzano Nmap per monitorare e gestire le proprie infrastrutture, ma anche per gli esperti di sicurezza, che lo impiegano per identificare potenziali punti deboli in una rete.

Il comando **nmap** funziona inviando pacchetti specifici a un dispositivo o a una rete e analizzando le risposte ricevute. Queste risposte forniscono informazioni cruciali sulla configurazione della rete, come l'indirizzo IP di un dispositivo, il tipo di sistema operativo, i servizi in esecuzione e le versioni di software che potrebbero essere vulnerabili. In pratica, quando si esegue un comando **nmap** su un indirizzo IP o un intervallo di indirizzi, Nmap invia pacchetti alle porte di quel dispositivo per vedere se rispondono e, se sì, quale tipo di servizio è in ascolto su quella porta.

Una delle caratteristiche distintive di **nmap** è la sua capacità di eseguire scansioni in vari modi, come scansioni di porte (per vedere quali porte sono aperte o chiuse), scansioni per determinare il sistema operativo di un host (OS fingerprinting), e scansioni per identificare applicazioni e versioni software. Può anche essere configurato per eseguire scansioni più avanzate, come l'analisi della vulnerabilità di una rete, ed è spesso utilizzato per scoprire le configurazioni errate e le potenziali falle di sicurezza in un ambiente di rete.

Inoltre, **nmap** è molto flessibile e supporta numerose opzioni per personalizzare la scansione in base alle necessità, permettendo di definire intervalli di porte, eseguire scansioni stealth (che cercano di minimizzare la possibilità di essere rilevati), o analizzare solo determinati tipi di traffico. La sua potenza e versatilità lo rendono uno strumento fondamentale per chiunque lavori nell'amministrazione di reti e sicurezza informatica. Tuttavia, a causa della sua capacità di eseguire scansioni approfondite, è uno strumento che può essere utilizzato anche in modo offensivo, per testare la sicurezza di una rete o per attacchi mirati, motivo per cui è importante che venga usato in modo etico e con permessi adeguati.

## Prova Pratica

Per esplorare tutte le funzionalità e le opzioni disponibili di **nmap**, è possibile digitare il comando `man nmap`. Questo comando apre la pagina di manuale di **nmap**, che fornisce una descrizione dettagliata delle varie opzioni e parametri che è possibile utilizzare con il programma. La pagina di manuale contiene informazioni su come eseguire diverse tipologie di scansioni, configurare le impostazioni avanzate e sfruttare al meglio le potenzialità di **nmap**. In pratica, il comando `man nmap` è una risorsa utile per approfondire la conoscenza di tutte le funzionalità offerte dallo strumento.

```
A typical Nmap scan is shown in Example 1. The only Nmap arguments used
in this example are -A, to enable OS and version detection, script
scanning, and traceroute; -T4 for faster execution; and then the
hostname.

Example 1. A representative Nmap scan

# nmap -A -T4 scanme.nmap.org

Nmap scan report for scanme.nmap.org (74.207.244.221)
Host is up (0.029s latency).
rDNS record for 74.207.244.221: li86-221.members.linode.com
Not shown: 995 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 5.3p1 Debian 3ubuntu7 (protocol 2.0)
|_ ssh-hostkey: 1024 8d:60:f1:7c:ca:b7:3d:0a:d6:67:54:9d:69:d9:b9:dd (DSA)
|_ 2048 79:f8:09:ac:d4:e2:32:42:10:49:d3:bd:20:82:85:ec (RSA)
80/tcp    open  http     Apache httpd 2.2.14 ((Ubuntu))
|_ http-title: Go ahead and ScanMe!
646/tcp   filtered ldap
1720/tcp  filtered H.323/Q.931
9929/tcp  open  nping-echo Nping echo
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6.39
OS details: Linux 2.6.39
Network Distance: 11 hops
Service Info: OS: Linux; CPE: cpe:/o:linux:kernel

TRACEROUTE (using port 53/tcp)
HOP RTT ADDRESS
[Cut first 10 hops for brevity]
11 17.65 ms li86-221.members.linode.com (74.207.244.221)

Nmap done: 1 IP address (1 host up) scanned in 14.40 seconds

The newest version of Nmap can be obtained from https://nmap.org. The
newest version of this man page is available at
https://nmap.org/book/man.html. It is also included as a chapter of
Nmap Network Scanning: The Official Nmap Project Guide to Network
Discovery and Security Scanning (see https://nmap.org/book/).
```

La prima scansione che effettueremo sarà sul nostro **localhost**, ovvero sulla macchina su cui stiamo eseguendo il comando. Il comando da utilizzare è **nmap -A -T4 localhost**. L'opzione **-A** abilita una scansione avanzata che include il rilevamento del sistema operativo, la versione dei servizi in esecuzione, il tracciamento delle tracce di rete (traceroute) e l'identificazione di eventuali script vulnerabili che potrebbero essere sfruttati. L'opzione **-T4** imposta la scansione su un livello di aggressività più elevato, che accelera il processo di scansione, riducendo al minimo i tempi di risposta ma mantenendo una buona accuratezza. Con questo comando, **nmap** eseguirà una scansione completa e dettagliata del nostro sistema locale.

```
[analyst@secOps ~]$ nmap -A -T4 localhost
Starting Nmap 7.70 ( https://nmap.org ) at 2024-12-13 10:41 EST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000020s 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 5
|   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
```

La scansione ci mostra le porte aperte e le relative versioni dei servizi attivi su tali porte.

Successivamente, eseguiremo una scansione dell'intera rete alla quale siamo connessi. Per farlo, utilizzeremo lo stesso comando visto in precedenza, ma invece di specificare **localhost**, inseriremo l'indirizzo IP della rete. Nel mio caso, l'indirizzo della rete è **10.0.2.0/24**, che indica un intervallo di indirizzi IP compreso tra **10.0.2.1** e **10.0.2.254**. Pertanto, il comando sarà **nmap -A -T4 10.0.2.0/24**. Questo eseguirà una scansione avanzata su tutti i dispositivi connessi alla rete, permettendoci di ottenere informazioni sui sistemi attivi, le porte aperte e i servizi in esecuzione all'interno dell'intervallo di indirizzi specificato. Tale operazione nel caso ci fossero diversi dispositivi connessi potrebbe richiedere svariati minuti per il proprio compimento.

Schermata che mostra lo scan dell'intera rete:

```
[analyst@secOps ~]$ nmap -A -T4 10.0.2.0/24
Starting Nmap 7.70 ( https://nmap.org ) at 2024-12-13 10:43 EST
Nmap scan report for 10.0.2.15
Host is up (0.000027s 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          0 Mar 26  2018 ftp_test
| ftp-syst:
|   STAT:
| FTP server status:
|   Connected to 10.0.2.15
|   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
```

Avendo solo un'Host all'interno della rete la scansione sarà identica alla precedente.

Infine, eseguiremo una scansione su un server esterno. È importante sottolineare che eseguire scansioni su server privati senza autorizzazione è illegale. Pertanto, è fondamentale scegliere server per i quali è esplicitamente consentito effettuare scansioni, come ad esempio **scanme.nmap.org**, un server messo a disposizione dagli sviluppatori di Nmap per scopi di test e apprendimento. In questo caso, il comando rimarrà invariato rispetto a quello utilizzato per la scansione della rete locale, ma al posto dell'indirizzo IP della rete, dovremo semplicemente inserire il nome del server, ossia **nmap -A -T4 scanme.nmap.org**. Questo comando eseguirà una scansione avanzata del server specificato, consentendoci di analizzare le sue porte aperte, i servizi in esecuzione e altre informazioni utili, sempre nel rispetto delle normative legali.

```
Terminal - analyst@secOps:-
File Edit View Terminal Tabs Help

[analyst@secOps ~]$ nmap -A -T4 scanme.nmap.org
Starting Nmap 7.70 ( https://nmap.org ) at 2024-12-13 10:38 EST
Stats: 0:00:39 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 99.82% done; ETC: 10:39 (0:00:00 remaining)
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:91ff:fe18:bb2f
Not shown: 996 filtered ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.13 (Ubuntu Linux; protocol 2.0)
| 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)
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
```

Come mostrano le schermate, **nmap** è uno strumento estremamente potente e versatile. In questo caso, abbiamo effettuato un'analisi approfondita di un server esterno, esaminando ogni dettaglio rilevante. Abbiamo ottenuto informazioni cruciali, come la versione del sistema operativo in uso, gli indirizzi IP associati al server e l'elenco delle porte aperte, con le relative versioni dei servizi in esecuzione su di esse. Questo ci ha permesso di ottenere una panoramica completa delle configurazioni e delle potenziali vulnerabilità del server, dimostrando l'efficacia di **nmap** nell'individuare dettagli cruciali sulla sicurezza di un sistema remoto.

## 5 - Analisi di un attacco Sql Injection

### Che cos'è un'attacco Sql Injection?

Un attacco **SQL Injection** è una tecnica di attacco informatico che sfrutta vulnerabilità nelle applicazioni web, in particolare quelle che interagiscono con database SQL. Questo tipo di attacco si verifica quando un'applicazione consente agli utenti di inviare input, come ad esempio attraverso moduli di login, barre di ricerca o altri campi interattivi, senza effettuare una corretta validazione o sanificazione dei dati inseriti. Gli aggressori possono quindi manipolare le query SQL inviate al database, iniettando comandi SQL dannosi all'interno di queste richieste.

Quando un'applicazione web è vulnerabile a un SQL Injection, l'attaccante può sfruttare il comportamento errato o incompleto del sistema per eseguire operazioni non autorizzate sul database. Queste operazioni possono includere la lettura di dati sensibili, la modifica o cancellazione di informazioni, l'accesso non autorizzato a dati riservati, o addirittura l'esecuzione di comandi sul sistema sottostante. In alcuni casi, l'iniezione di codice SQL può anche permettere all'attaccante di ottenere il controllo totale dell'intero database, compromettendo gravemente la sicurezza dell'applicazione e dei dati che essa gestisce.

Le vulnerabilità di **SQL Injection** derivano spesso da una gestione inadeguata dei dati provenienti dall'utente. In un'applicazione vulnerabile, l'input dell'utente viene concatenato direttamente nelle query SQL, senza alcuna verifica che i dati siano sicuri. Questo consente agli aggressori di manipolare la logica della query SQL in modi imprevisi, aggirando i controlli di autenticazione e autorizzazione o inducendo il sistema a restituire informazioni riservate.

La gravità di un attacco **SQL Injection** dipende dalle capacità dell'aggressore e dalle misure di sicurezza implementate nel sistema target. Se il database è configurato in modo errato o se non vengono utilizzate pratiche di sicurezza adeguate, come l'uso di query parametrizzate o la validazione rigorosa dell'input, l'attacco può avere conseguenze devastanti. Questo include il furto di dati, l'accesso non autorizzato a sistemi aziendali o la compromissione di interi sistemi informatici.

Per prevenire gli attacchi di SQL Injection, è fondamentale che le applicazioni adottino misure di sicurezza efficaci, come l'uso di query parametrizzate (dove i dati dell'utente vengono trattati come parametri separati e non come parte del codice SQL), la validazione e la sanificazione dei dati in ingresso, nonché il restringimento dei privilegi di accesso al database, per limitare i danni in caso di compromissione.

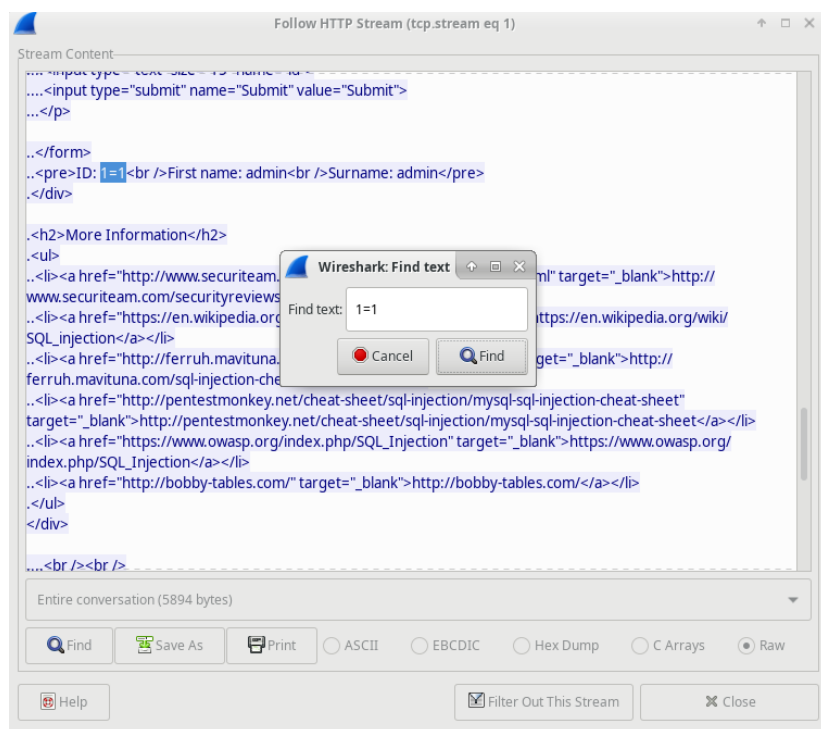


## Prova Pratica

In questo caso, andrò ad analizzare un file già esistente sulla macchina **cyberops**, chiamato **sql\_lab.pcap**, che si trova nella directory **/home/analyst/lab.support.files**. Dopo aver aperto il file con uno strumento come Wireshark, troverò una cattura che dura più di 8 minuti, corrispondente alla durata dell'attacco. Dalla cattura, si nota immediatamente che gli indirizzi IP coinvolti nell'attacco sono **10.0.2.4** e **10.0.2.15**.

La prima riga che andrò ad esaminare è la riga **13**. Per analizzare meglio la richiesta HTTP in questa riga, farò clic destro sulla voce interessata e, nel menu a tendina, selezionerò l'opzione **"HTTP Stream"**. Questa operazione consente di visualizzare l'intera comunicazione HTTP tra il client e il server relativa a quella specifica richiesta, facilitando l'analisi dei dati trasmessi.

Un **HTTP Stream** è una funzionalità che permette di visualizzare, in modo continuo, tutti i pacchetti che appartengono alla stessa sessione HTTP. In pratica, raggruppa tutte le richieste e risposte HTTP in un'unica sequenza comprensibile, mostrando la comunicazione completa tra il client e il server. In questo modo, è possibile seguire il flusso delle informazioni, esaminare i dettagli della richiesta, come i parametri inviati, le intestazioni e i contenuti, senza dover analizzare singolarmente ogni pacchetto. Questo è particolarmente utile per tracciare attività come un attacco di **SQL Injection**, dove si cerca di analizzare come i dati vengono inviati e ricevuti tra il browser dell'attaccante e il server vulnerabile.



Questa è la finestra che avremo a disposizione una volta aperto il **HTTP Stream**. In questa visualizzazione, possiamo successivamente filtrare il contenuto per cercare una parte di testo specifica. La porzione di testo che ci interessa in questo caso è una richiesta che potrebbe essere stata inviata al database, come ad esempio **'1=1'**.

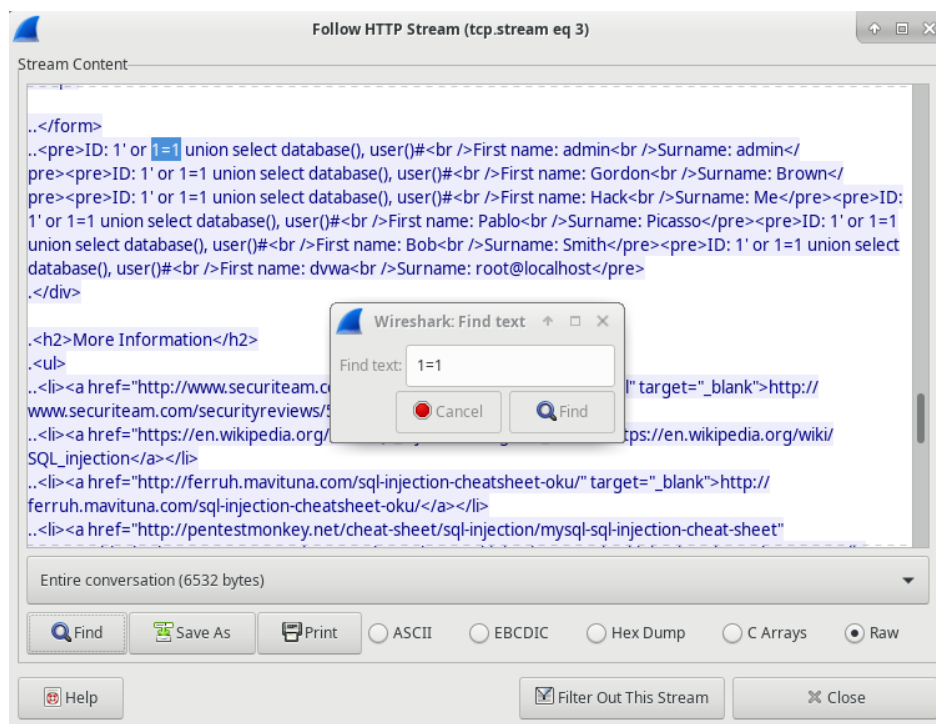
Una richiesta SQL come **'1=1'** è un esempio di una tecnica comunemente utilizzata negli attacchi di **SQL Injection**. In questo tipo di attacco, l'aggressore manipola la query SQL inviata al database, inserendo condizioni che sono sempre vere, come appunto **'1=1'**.

Questo tipo di espressione è sempre vera (poiché 1 è sempre uguale a 1), e viene utilizzato per aggirare i controlli di autenticazione, o per estrarre informazioni dal database che altrimenti sarebbero protette. Ad esempio, in un contesto di login, un attaccante potrebbe utilizzare una query SQL manipolata in modo che **'1=1'** venga aggiunto alla condizione di autenticazione, forzando il sistema a restituire un risultato positivo, permettendo così l'accesso non autorizzato.

Quando cerchiamo **'1=1'** nel traffico HTTP, stiamo cercando un'indicazione di questo tipo di manipolazione, che potrebbe suggerire l'intenzione di un attacco di SQL Injection. Il traffico potrebbe contenere questo tipo di stringa come parte di una richiesta HTTP inviata al server, tentando di sfruttare una vulnerabilità nella gestione delle query SQL.

In questo caso, l'aggressore sta cercando di determinare se il server è vulnerabile a un attacco di **SQL Injection**. Per fare ciò, invia una richiesta manipolata, come ad esempio l'inserimento di una condizione sempre vera, come **'1=1'**, all'interno della query SQL. L'obiettivo è verificare se il server risponde in modo anomalo, indicando che la query SQL non è correttamente sanitizzata e che il sistema potrebbe essere vulnerabile a manipolazioni esterne. Se il server accetta e elabora la richiesta senza errori, ciò potrebbe confermare la vulnerabilità del sistema al tipo di attacco **SQL Injection**.

La seconda riga che andremo ad analizzare sarà la riga 19. Lo faremo sempre con lo stesso metodo.

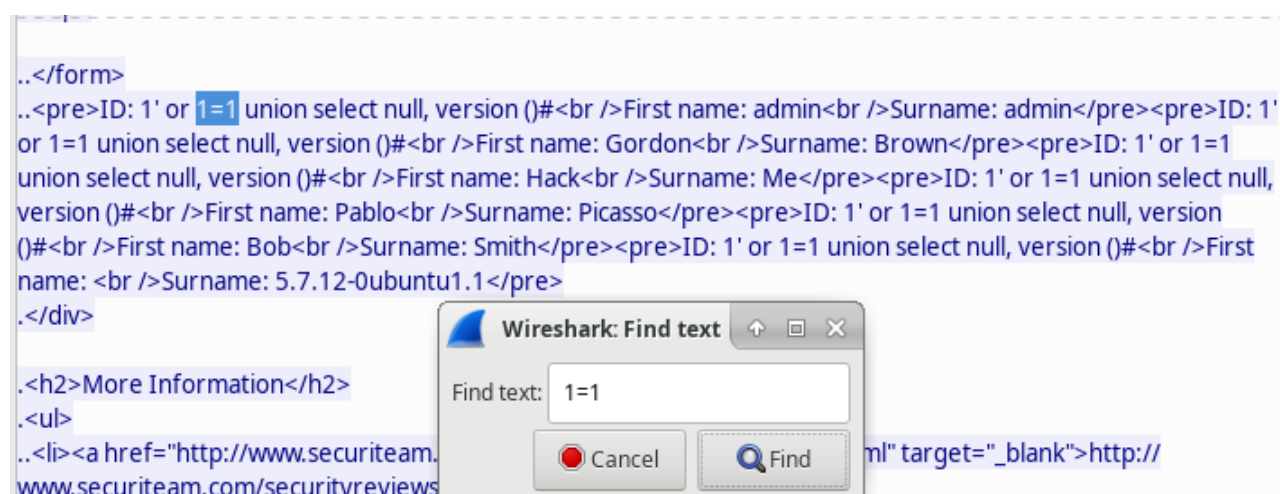


In questo caso, la query utilizzata dall'aggressore è più complessa e prende la forma di **'1' or 1=1 union select database, user()#**. Questa è una forma avanzata di **SQL Injection** che tenta di sfruttare diverse tecniche per ottenere informazioni dal database. La parte **'1' or 1=1** è un tentativo di manipolare la logica della query, creando una condizione sempre vera, come nel caso precedente, per aggirare eventuali controlli di accesso o autenticazione.

La clausola **'union select** viene utilizzata per combinare il risultato della query originale con una nuova query che restituisce dati dal database. In questo caso, **union select database, user()** cerca di estrarre il nome del database corrente e l'utente che sta eseguendo la query sul server.

Infine, il simbolo **#** viene usato per commentare il resto della query, assicurandosi che il database ignori eventuali parti della query originale che potrebbero causare errori o interferire con l'iniezione. In questo modo, l'aggressore cerca di ottenere informazioni sensibili sul database, come il nome del database e l'utente, senza essere rilevato.

Successivamente troveremo un altro riscontro alla riga 22:

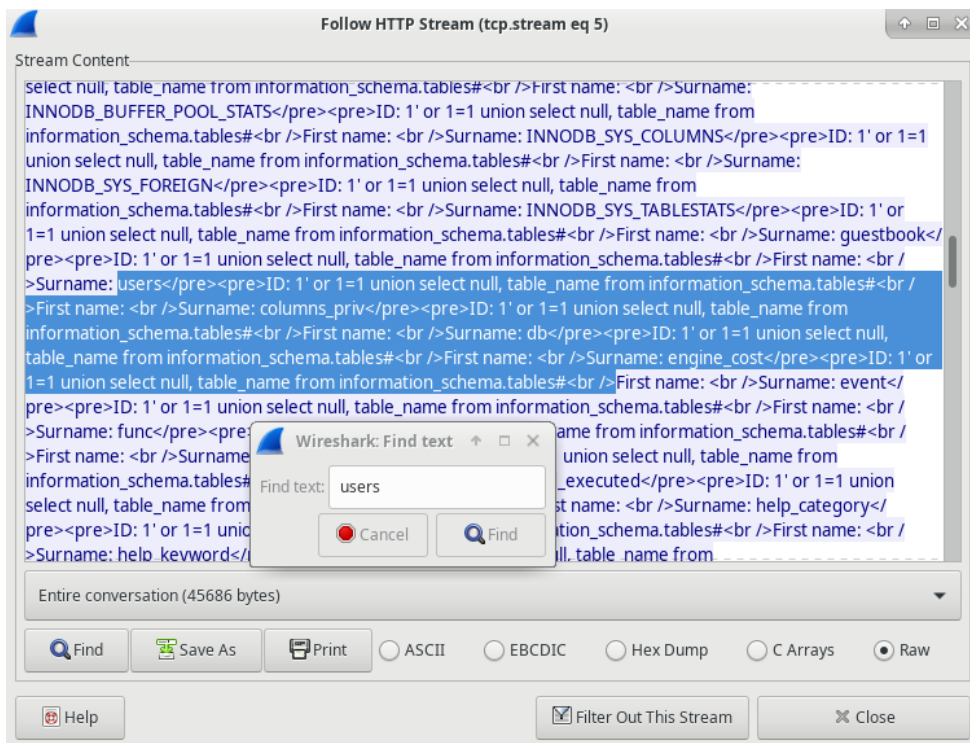


In questa riga la Query è **1' or 1=1 union select null, version()#**. La parte **'union select null, version()'** della query SQL serve a combinare il risultato della query originale con i dati di una seconda query. In questo caso, **null** viene inserito per riempire una colonna vuota, mentre **version()** è una funzione che restituisce la versione del database in uso. L'intento dell'attaccante è ottenere informazioni sulla versione del database, che potrebbero rivelare eventuali vulnerabilità sfruttabili per un attacco.

La quarta riga che analizzerò sarà la riga 25, dove utilizzerò il filtro **'users'**. La query in questione è **'1' or 1=1 union select null, table\_name from information\_schema.tables#'**. A partire dalla parte **'union select null, table\_name'**, questa query cerca di unire il risultato della query originale con i dati provenienti da una seconda query. Qui, **null** è usato per riempire una colonna, mentre **table\_name** è una colonna che contiene i nomi delle tabelle nel database.

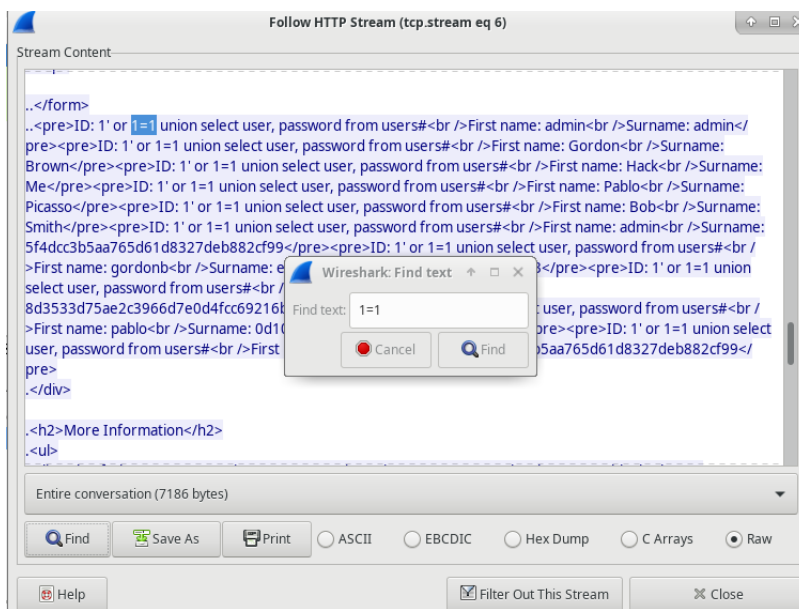
La query accede alla vista **information\_schema.tables**, che è una parte del database che contiene informazioni su tutte le tabelle presenti nel sistema. L'obiettivo dell'attaccante, quindi, è ottenere un elenco delle tabelle presenti nel database. Il simbolo **#** alla fine della query è un commento che fa sì che il resto della query venga ignorato, impedendo che eventuali errori nella sintassi causino problemi nell'esecuzione della query. In sostanza,

l'attaccante sta cercando di scoprire la struttura del database, raccogliendo i nomi delle tabelle per pianificare ulteriori azioni.



L'ultima riga che andrò ad analizzare sarà la riga 28. La query utilizzata è **'1' or 1=1 union select user, password from users#'**. In questa parte dell'attacco, l'aggressore cerca di ottenere informazioni sensibili, in particolare **i nomi utente e le password** degli utenti salvati nel database. La query **'union select user, password from users'** unisce il risultato della query originale con i dati estratti dalla tabella **users**, che contiene le credenziali degli utenti, più precisamente **i nomi utente (user)** e le relative **password** (o, più probabilmente, gli **hash delle password**).

L'intento dell'attaccante, utilizzando questa query, è di estrarre i nomi utente e gli hash delle password memorizzati nella tabella **users** del database. In pratica, il comando cerca di bypassare i controlli di accesso, eseguendo una **SQL Injection** per raccogliere informazioni che potrebbero essere utilizzate per compiere un attacco successivo, come il furto di credenziali o l'accesso non autorizzato al sistema.



Nel caso specifico, se l'hash della password 8d3533d75ae2c3966d7e0d4fcc69216b è stato ottenuto, l'attaccante potrebbe successivamente utilizzare uno strumento online come **<https://crackstation.net/>** per decrittare l'hash e scoprire la password in chiaro. In questo caso, l'hash 8d3533d75ae2c3966d7e0d4fcc69216b corrisponde alla password in chiaro **charley**, che potrebbe appartenere a un utente del sistema, come ad esempio l'utente 1337.

Questo processo mostra come un attacco di **SQL Injection** possa portare all'accesso a credenziali sensibili, che poi possano essere utilizzate per ulteriori attacchi o per ottenere il controllo del sistema compromesso.

## 6 - Conclusioni

I diversi laboratori svolti oggi ci hanno permesso di esaminare in dettaglio le informazioni sui sistemi Windows e Linux, ma ci hanno anche mostrato come sia relativamente semplice per un attaccante ottenere accesso a dati sensibili. Un attaccante potrebbe, infatti, eseguire una scansione della rete per identificare i dispositivi vulnerabili, recuperare informazioni non protette tramite il protocollo HTTP, oppure sfruttare vulnerabilità come la SQL Injection per estrarre dati sensibili dal database.

Per limitare queste minacce, è fondamentale implementare misure di sicurezza adeguate. Ad esempio, per proteggere le comunicazioni da attacchi di tipo "man-in-the-middle" e per evitare che i dati sensibili siano trasmessi in chiaro, è indispensabile utilizzare sempre HTTPS al posto di HTTP, garantendo così che le informazioni siano criptate durante il loro transito. Inoltre, per evitare che gli attaccanti possano eseguire scansioni della rete e scoprire porte aperte o servizi vulnerabili, è consigliabile configurare un firewall robusto che limiti l'accesso esterno e monitori costantemente il traffico di rete.

Inoltre, per proteggere i sistemi da attacchi di SQL Injection, è essenziale adottare pratiche di programmazione sicure, come l'uso di query parametrizzate, che impediscono l'iniezione di comandi maligni all'interno delle query SQL. È anche importante mantenere sempre aggiornati i sistemi operativi e le applicazioni, così da ridurre il rischio di sfruttamento di vulnerabilità note. Infine, per garantire una protezione aggiuntiva, l'uso di strumenti di monitoraggio e auditing del traffico di rete, come Wireshark, può essere utile per rilevare attività sospette e rispondere prontamente a potenziali minacce.