

— **Analisi avanzate: Un approccio pratico**

Laboratorio

Indice

-
- Laboratorio - Utilizzo di Windows PowerShell
 - Laboratorio - Utilizzo di Wireshark per Esaminare il Traffico HTTP e HTTPS

Bonus:

- Laboratorio - Esplorazione di Nmap
- Attacco a un Database MySQL

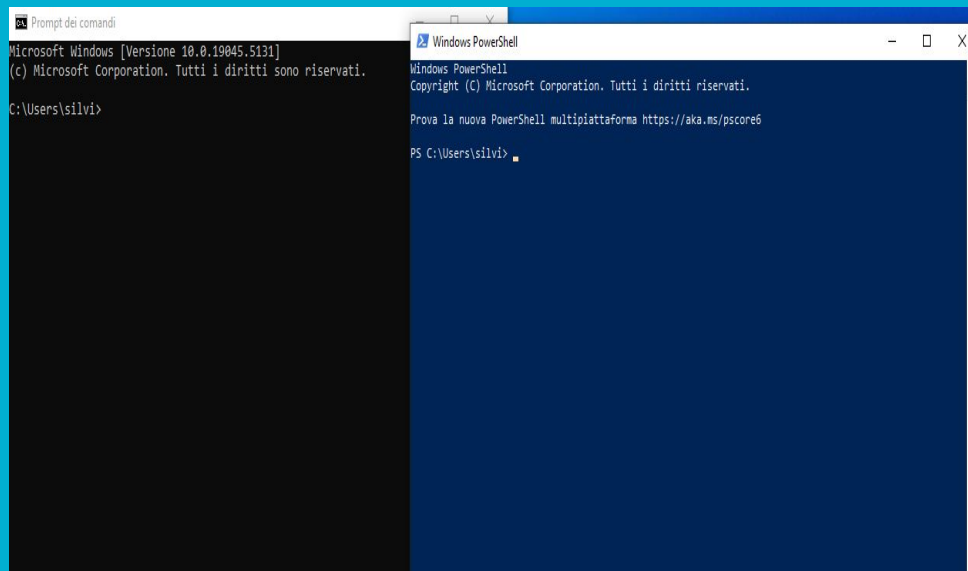
Laboratorio - Utilizzo di Windows PowerShell

Obiettivi

- **Parte 1: accedere alla console di PowerShell.**
- **Parte 2: Esplora i comandi del prompt dei comandi e di PowerShell.**
- **Parte 3: Esplora i cmdlet.**
- **Parte 4: Esplora il comando netstat utilizzando PowerShell.**
- **Parte 5: Svuotare il cestino tramite PowerShell.**

Parte 1: accedere alla console di PowerShell.

PowerShell e Prompt dei comandi sono strumenti di Windows che permettono di interagire con il sistema operativo attraverso comandi testuali. Il Prompt dei comandi, più semplice e basato sull'architettura MS-DOS, è utilizzato principalmente per operazioni di base come la gestione di file e directory. PowerShell, invece, è uno strumento più avanzato e moderno, progettato per amministratori di sistema, con un potente linguaggio di scripting e supporto per l'automazione di attività complesse. Mentre il Prompt dei comandi si limita a comandi tradizionali, PowerShell consente l'utilizzo di cmdlet e script per la gestione avanzata di risorse locali e di rete.



Parte 2: Esplora i comandi del prompt dei comandi e di PowerShell.

Quando si utilizza il comando `dir` sia nel Prompt dei comandi sia in PowerShell, entrambi mostrano un elenco di sottodirectory e file presenti nella directory corrente, includendo informazioni come tipo, dimensione, data e ora dell'ultima modifica. Tuttavia, PowerShell aggiunge dettagli sugli attributi/modalità dei file, offrendo un output leggermente più ricco.

Eseguendo altri comandi come `ping`, `cd` o `ipconfig`, i risultati sono simili in entrambe le finestre, poiché questi comandi sono comuni e vengono eseguiti allo stesso modo in entrambi gli ambienti. Nonostante ciò, PowerShell può gestire una gamma più ampia di comandi e script avanzati rispetto al Prompt dei comandi, mostrando la sua maggiore versatilità.

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

Prova la nuova PowerShell multiplatforma https://aka.ms/pscore6

PS C:\Users\silvi> dir

Directory: C:\Users\silvi

Mode                LastWriteTime         Length Name
----                -
d-----          05/12/2024      09:42            3D Objects
d-----          05/12/2024      09:42            Contacts
d-----         10/12/2024     14:06            Desktop
d-----          05/12/2024      09:42            Documents
d-----         10/12/2024     14:06            Downloads
d-----          05/12/2024      09:42            Favorites
d-----          05/12/2024      09:42            Links
d-----          05/12/2024      09:42            Music
d-----          05/12/2024     11:59            OneDrive
d-----          05/12/2024      09:44            Pictures
d-----          05/12/2024      09:42            Saved Games
d-----          05/12/2024      09:44            Searches
d-----         10/12/2024     14:47            Videos

PS C:\Users\silvi>
```

```
Prompt dei comandi
Microsoft Windows [Versione 10.0.19045.5131]
(c) Microsoft Corporation. Tutti i diritti sono riservati.

C:\Users\silvi>dir
Il volume nell'unità C non ha etichetta.
Numero di serie del volume: 78D5-945E

Directory di C:\Users\silvi
10/12/2024  14:02  <DIR>          .
10/12/2024  14:02  <DIR>          ..
05/12/2024  09:42  <DIR>          3D Objects
05/12/2024  09:42  <DIR>          Contacts
10/12/2024  14:06  <DIR>          Desktop
05/12/2024  09:42  <DIR>          Documents
10/12/2024  14:06  <DIR>          Downloads
05/12/2024  09:42  <DIR>          Favorites
10/12/2024  14:05  <DIR>          Links
05/12/2024  09:42  <DIR>          Music
05/12/2024  09:42  <DIR>          OneDrive
05/12/2024  09:42  <DIR>          Pictures
05/12/2024  09:44  <DIR>          Saved Games
06/12/2024  11:59  <DIR>          Searches
05/12/2024  09:44  <DIR>          Videos
05/12/2024  09:42  <DIR>          .
05/12/2024  09:42  <DIR>          ..
05/12/2024  09:44  <DIR>          Desktop
06/12/2024  11:59  <DIR>          Documents
05/12/2024  09:44  <DIR>          Downloads
05/12/2024  09:42  <DIR>          Favorites
05/12/2024  09:42  <DIR>          Links
05/12/2024  09:42  <DIR>          Music
06/12/2024  11:59  <DIR>          OneDrive
05/12/2024  09:44  <DIR>          Pictures
05/12/2024  09:42  <DIR>          Saved Games
05/12/2024  09:44  <DIR>          Searches
10/12/2024  14:47  <DIR>          Videos
0 File                                0 byte
15 Directory 23.160.168.448 byte disponibili

C:\Users\silvi>
```

dir

ping

```
PS C:\Users\silvi> 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=27ms TTL=115
Risposta da 8.8.8.8: byte=32 durata=27ms TTL=115
Risposta da 8.8.8.8: byte=32 durata=26ms TTL=115
Risposta da 8.8.8.8: byte=32 durata=27ms TTL=115
```

```
Statistiche Ping per 8.8.8.8:
    Pacchetti: Trasmessi = 4, Ricevuti = 4,
    Persi = 0 (0% persi),
    Tempo approssimativo percorsi andata/ritorno in millisecondi:
        Minimo = 26ms, Massimo = 27ms, Medio = 26ms
PS C:\Users\silvi>
```

```
C:\Users\silvi>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=26ms TTL=115
Risposta da 8.8.8.8: byte=32 durata=26ms TTL=115
Risposta da 8.8.8.8: byte=32 durata=27ms TTL=115
Risposta da 8.8.8.8: byte=32 durata=26ms TTL=115
```

```
Statistiche Ping per 8.8.8.8:
    Pacchetti: Trasmessi = 4, Ricevuti = 4,
    Persi = 0 (0% persi),
    Tempo approssimativo percorsi andata/ritorno in millisecondi:
        Minimo = 26ms, Massimo = 27ms, Medio = 26ms
```

```
Scheda Ethernet Ethernet:
```

```
Suffisso DNS specifico per connessione:
Indirizzo IPv6 locale rispetto al collegamento . : fe80::fb34:dc82:719d:ee50%13
Indirizzo IPv4. . . . . : 192.168.1.7
Subnet mask . . . . . : 255.255.255.0
Gateway predefinito . . . . . : 192.168.1.1
PS C:\Users\silvi>
```

```
Scheda Ethernet Ethernet:
```

```
Suffisso DNS specifico per connessione:
Indirizzo IPv6 locale rispetto al collegamento . : fe80::fb34:dc82:719d:ee50%13
Indirizzo IPv4. . . . . : 192.168.1.7
Subnet mask . . . . . : 255.255.255.0
Gateway predefinito . . . . . : 192.168.1.1
C:\Users\silvi>
```

ipconfig

Parte 3: Esplora i cmdlet.

In PowerShell, i cmdlet (comandi predefiniti) seguono una convenzione di denominazione basata sullo schema **Verbo-Nome** (es. `Get-ChildItem`). Per esempio, il comando PowerShell equivalente al comando `dir` del Prompt dei comandi è `Get-ChildItem`, come mostrato dall'immagine. Questo approccio standardizzato migliora la leggibilità e la comprensione dei comandi.

Mentre `dir` è un alias usato per compatibilità con il Prompt dei comandi, il cmdlet vero e proprio (`Get-ChildItem`) offre maggiore flessibilità, consentendo l'uso di parametri avanzati. I cmdlet di PowerShell sono potenti strumenti progettati per attività complesse, come la gestione di file, configurazioni di sistema e risorse di rete.

Al termine dell'analisi, è sufficiente chiudere la finestra del Prompt dei comandi o di PowerShell.

Get-Alias dir

```
PS C:\Users\silvi> Get-Alias dir
```

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

```
PS C:\Users\silvi>
```

Parte 4: Esplora il comando netstat utilizzando PowerShell.

Il comando **netstat**, utilizzabile sia nel Prompt dei comandi sia in PowerShell, permette di analizzare connessioni di rete, tabelle di routing e processi associati.

1. **Esplorazione delle opzioni:** Utilizzando il comando **netstat -h**, è possibile visualizzare tutte le opzioni disponibili. Queste includono parametri per visualizzare connessioni attive, statistiche di rete e tabelle di routing.
2. **Tabella di routing:** Il comando **netstat -r** o **netstat -ra** mostra la tabella di routing con percorsi attivi. Ad esempio, l'**IPv4 Gateway** rappresenta il punto di accesso alla rete esterna ed è spesso identificato dall'indirizzo **192.168.1.1**.
3. **Connessioni TCP e processi:** Utilizzando il comando **netstat -abno** in PowerShell con privilegi elevati, si ottengono informazioni dettagliate sulle connessioni TCP attive, inclusi i numeri di **PID** (Process ID) associati. Il PID consente di identificare il processo responsabile della connessione.
4. **Analisi tramite Task Manager:** Nel **Task Manager**, la scheda **Dettagli** permette di ordinare i processi per **PID**. Cliccando con il tasto destro su un PID specifico, è possibile accedere alla finestra **Proprietà**, che fornisce dettagli come il nome del servizio, l'utilizzo di memoria (ad esempio **5.804K**) e altre informazioni sul processo.

Questa procedura consente di associare connessioni di rete a processi specifici, utile per identificare attività sospette o analizzare l'uso delle risorse del sistema.

```

C:\Users\stlivi> netstat -h

Visualizza le statistiche del protocollo e le connessioni di rete TCP/IP correnti.

NETSTAT [-a] [-b] [-e] [-f] [-n] [-o] [-p >processo<] [-r] [-s] [-t] [-x] [-y] [-interval]

- a Visualizza tutte le connessioni e le porte di ascolto.
- b Visualizza l'eseguibale coinvolto nella creazione di ogni connessione o
  porta di ascolto. In alcuni casi, host di rete eseguiscono molti
  più componenti indipendenti e in questi casi il
  sequenza di componenti coinvolti nella creazione della connessione
  non è la porta in ascolto. In questo caso, l'eseguibale
  il nome e [t] nella porta inferiore, in alto è il componente che ha chiamato,
  e così via fino al raggiungimento di TCP/IP. Si noti che questa opzione
  può richiedere molto tempo e avrà esito negativo, a meno che non siano sufficienti
  autorizzazioni.
- e Visualizza le statistiche Ethernet. È possibile combinare
  opzione.
- r Visualizza nomi di dominio completi (FQDN) per stranieri
  indirizzi.
- s Visualizza indirizzi e numeri di porta in formato numerico.
- t Visualizza l'ID del processo proprietario associato a ogni connessione.
- p >processo< Mostra le connessioni per il protocollo specificato da >processo<;
  >processo< può essere qualsiasi: TCP, UDP, TCPv6 o UDPv6. Se usato
  con l'opzione -a, la visualizzazione delle statistiche per protocollo,
  >processo< può essere qualsiasi:
  IP, IPv6, ICMP, ICMPv6, TCP, TCPv6, UDP o UDPv6.
- q Visualizza tutte le connessioni, le porte di ascolto e i binding
  non in ascolto di porta TCP. Le porte di nonlistening associate possono o meno essere
  associate a una connessione attiva.
- n Visualizza la tabella di routing.
- s Visualizza le statistiche per protocollo. Per impostazione predefinita, le statistiche vengono
  visualizzate per IPv4, IPv6, ICMP, ICMPv6, TCP, TCPv6, UDP, UDPv6. Se usato
  con l'opzione -a, può essere utilizzata per specificare un sottoinsieme del valore predefinito.
- t Visualizza lo stato corrente di offload della connessione.
- x Visualizza connessioni NetworkDirect, listener e condivisi
  endpoint.
- y Visualizza il modello di connessione TCP per tutte le connessioni.
  Non può essere combinato con le altre opzioni.

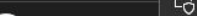
```

[illegible]

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

```
Connection active
```

Proto	Indirizzo locale	Indirizzo esterno	Stato	PID
TCP	0.0.0.0:135	0.0.0.0:0	LISTENING	968
Rpcss				
(svchost.exe)				
TCP	0.0.0.0:445	0.0.0.0:0	LISTENING	4
Impossibile ottenere informazioni sulla proprietà				
TCP	0.0.0.0:5040	0.0.0.0:0	LISTENING	984
CDPSvc				
(svchost.exe)				
TCP	0.0.0.0:5357	0.0.0.0:0	LISTENING	4
Impossibile ottenere informazioni sulla proprietà				
TCP	0.0.0.0:7060	0.0.0.0:0	LISTENING	4832
Impossibile ottenere informazioni sulla proprietà				
TCP	0.0.0.0:49664	0.0.0.0:0	LISTENING	728
(lsass.exe)				
TCP	0.0.0.0:49665	0.0.0.0:0	LISTENING	568
Impossibile ottenere informazioni sulla proprietà				
TCP	0.0.0.0:49666	0.0.0.0:0	LISTENING	1228
Eventlog				
(svchost.exe)				
TCP	0.0.0.0:49667	0.0.0.0:0	LISTENING	1580
Schedule				
(svchost.exe)				
TCP	0.0.0.0:49669	0.0.0.0:0	LISTENING	2948
(spoolsv.exe)				
TCP	0.0.0.0:49670	0.0.0.0:0	LISTENING	728
(lsass.exe)				
TCP	0.0.0.0:49671	0.0.0.0:0	LISTENING	704
Impossibile ottenere informazioni sulla proprietà				
TCP	192.168.1.7:139	0.0.0.0:0	LISTENING	4
Impossibile ottenere informazioni sulla proprietà				
TCP	192.168.1.7:49747	20.54.37.73:443	ESTABLISHED	3268
UpdService				
(svchost.exe)				
TCP	192.168.1.7:49770	151.5.18.105:80	CLOSE_WAIT	7484
(SmanSvc.exe)				



Nome	PID	Stato	Nome utente	CPU	Memoria (...)	Virtualizza...
Taskmgr.exe	3728	In esecuzione	silvi	00	18.804 K	Noi
svchost.exe	3860	In esecuzione	SERVIZIO LOCALE	00	1.572 K	Noi
dllhost.exe	3940	In esecuzione	SYSTEM	00	1.116 K	Noi
MicrosoftEdgeUpdat...	3952	In esecuzione	SYSTEM	00	864 K	Noi
svchost.exe	4116	In esecuzione	SYSTEM	00	2.220 K	Noi
SgmnBroker.exe	4224	In esecuzione	SYSTEM	00	2.700 K	Noi
svchost.exe	4260	In esecuzione	SERVIZIO LOCALE	00	1.312 K	Noi
taskhostw.exe	4312	In esecuzione	silvi	00	2.488 K	Dis...
svchost.exe	4424	In esecuzione	SYSTEM	00	880 K	Noi
svchost.exe	4456	In esecuzione	SERVIZIO LOCALE	00	1.492 K	Noi
svchost.exe	4488	In esecuzione	SYSTEM	00	972 K	Noi
AggregatorHost.exe	4564	In esecuzione	SYSTEM	00	520 K	Noi
svchost.exe	4592	In esecuzione	silvi	00	5.016 K	Dis...
msedgeview2.exe	4728	Sospeso	silvi	00	0 K	Dis...
svchost.exe	4832	In esecuzione	SERVIZIO DI RETE	00	5.804 K	Noi
NisSrv.exe	4888	In esecuzione	SERVIZIO LOCALE	00	2.824 K	Noi
ctfmon.exe	4932	In esecuzione	silvi	00	3.412 K	Dis...
svchost.exe	5156	In esecuzione	SYSTEM	00	900 K	Noi
SystemSettings.exe	5172	Sospeso	silvi	00	0 K	Dis...
TextInputHost.exe	5244	In esecuzione	silvi	00	4.556 K	Dis...
msedgeview2.exe	5416	Sospeso	silvi	00	0 K	Dis...
StartMenuExperienc...	5448	In esecuzione	silvi	00	15.844 K	Dis...

Parte 5: Svuotare il cestino tramite PowerShell.

PowerShell consente di eseguire in modo rapido e automatizzato operazioni che normalmente richiederebbero più passaggi nell'interfaccia grafica di Windows. Un esempio pratico è lo svuotamento del Cestino tramite il comando **Clear-RecycleBin**.

1. **Preparazione:** Prima di eseguire il comando, si verifica che ci siano file nel Cestino. Se non presenti, si creano file (ad esempio, con Blocco Note) e li si elimina per posizionarli nel Cestino.
2. **Esecuzione del comando:** Digitando **Clear-RecycleBin** in PowerShell, i file presenti nel Cestino vengono eliminati definitivamente dal sistema, senza la necessità di passare attraverso i menu grafici di Windows.
3. **Risultato:** Al termine dell'operazione, il Cestino risulta completamente svuotato. Questa azione è utile in ambienti di rete o su più dispositivi, dove l'automazione tramite script PowerShell consente di risparmiare tempo e standardizzare le operazioni.

Questo comando dimostra come PowerShell possa semplificare e velocizzare attività amministrative, anche su larga scala, rispetto all'interfaccia grafica.

```
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"): Sì
[S] Sì [T] Sì a tutti [N] No [U] No a tutti [O] Sospendi [?] Guida (il valore predefinito è "S"): s
PS C:\Windows\system32>
```

Clear-RecycleBin

Conclusione

L'utilizzo di PowerShell dimostra come strumenti avanzati basati su comandi possano semplificare e ottimizzare attività amministrative su sistemi Windows. Dalla gestione di file e processi di rete all'automazione di operazioni come lo svuotamento del Cestino, PowerShell offre una flessibilità e un controllo maggiori rispetto agli strumenti grafici tradizionali. La sua potenza risiede nei cmdlet standardizzati, nell'abilità di gestire script complessi e nell'applicazione su più dispositivi, rendendolo uno strumento essenziale per amministratori di sistema e utenti esperti.

Laboratorio - Utilizzo di Wireshark per Esaminare il Traffico HTTP e HTTPS

Obiettivi

- **Parte 1: Cattura e visualizza il traffico HTTP**
- **Parte 2: Cattura e visualizza il traffico HTTPS**

Parte 1: Cattura e visualizza il traffico HTTP

Nel contesto di un esercizio di monitoraggio del traffico di rete, è stato utilizzato **tcpdump** per catturare i pacchetti HTTP su una macchina virtuale configurata con il sistema operativo Linux. L'obiettivo era quello di catturare e analizzare il traffico di rete tra un browser web e un sito HTTP non crittografato, utilizzando strumenti come **tcpdump** e **Wireshark**.

Il processo è iniziato con l'apertura di un terminale e l'esecuzione del comando **tcpdump -i enp0s3 -s 0 -w httpdump.pcap**, che ha catturato il traffico HTTP sull'interfaccia di rete **enp0s3**. I parametri del comando specificano di catturare tutti i pacchetti, senza limitazioni sulla dimensione dei pacchetti, e di salvare i dati in un file chiamato **httpdump.pcap**. Questo comando ha permesso di registrare il traffico mentre il browser web veniva utilizzato per navigare su una pagina HTTP del sito <http://www.altoromutual.com/login.jsp>.

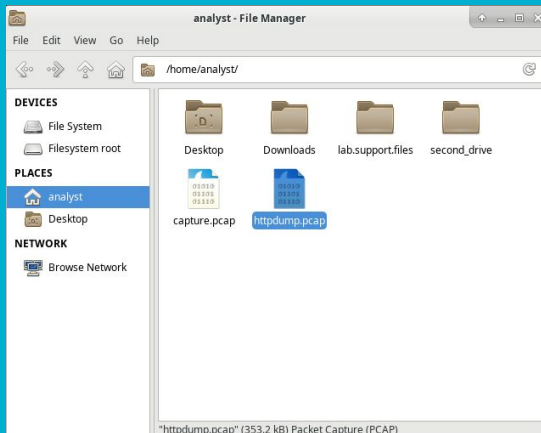
Dopo aver interagito con il sito (inserendo un nome utente e una password), il traffico HTTP è stato catturato fino a quando non è stato interrotto con **CTRL+C**. Successivamente, il file **httpdump.pcap** è stato aperto con **Wireshark** per l'analisi. All'interno di Wireshark, è stato applicato un filtro per visualizzare solo i messaggi HTTP, esaminando i dettagli dei pacchetti.

Esaminando uno dei pacchetti **POST** inviati al server, è stato possibile osservare informazioni cruciali nel corpo del messaggio, in particolare nell'area **HTML Form URL Encoded**. Due dati sensibili sono stati visualizzati: **UID dell'amministratore** e **password dell'amministratore**. Questo esempio evidenzia come il traffico HTTP non crittografato possa esporre informazioni sensibili durante la trasmissione, rendendo vulnerabili i dati degli utenti.

In sintesi, questa attività ha dimostrato l'importanza di monitorare il traffico di rete, ma anche il rischio di esposizione dei dati quando il traffico non è adeguatamente protetto, come nel caso di connessioni HTTP non sicure.

Parte 1: Cattura e visualizza il traffico HTTP

```
Terminal - analyst@secOps:~  
File Edit View Terminal Tabs Help  
[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  
^C974 packets captured  
974 packets received by filter  
0 packets dropped by kernel  
[analyst@secOps ~]$
```



No.	Time	Source	Destination	Protocol	Length	Info
218	1.746247	10.0.2.15	65.61.137.117	HTTP	406	GET /images/logo.gif HTTP/1.1
524	17.607349	10.0.2.15	65.61.137.117	HTTP	403	GET /images/logo.gif HTTP/1.1
525	17.607415	10.0.2.15	65.61.137.117	HTTP	406	GET /images/logo.gif HTTP/1.1
530	17.615493	10.0.2.15	65.61.137.117	HTTP	400	GET /images/logo.gif HTTP/1.1
532	17.620767	65.61.137.117	10.0.2.15	HTTP	1175	HTTP/1.1 200 OK (PNG) [PNG image]
542	17.707246	65.61.137.117	10.0.2.15	HTTP	354	HTTP/1.1 200 OK (GIF89a)
546	17.774613	65.61.137.117	10.0.2.15	HTTP	2391	HTTP/1.1 200 OK (GIF89a)
548	17.911642	65.61.137.117	10.0.2.15	HTTP	3354	HTTP/1.1 200 OK (PNG) [PNG image]
554	17.916113	10.0.2.15	65.61.137.117	HTTP	404	GET /favicon.ico HTTP/1.1
556	17.933559	10.0.2.15	65.61.137.117	HTTP	348	GET /favicon.ico HTTP/1.1
564	18.083935	65.61.137.117	10.0.2.15	HTTP	1648	HTTP/1.1 404 Not Found (text/html)
578	18.083935	10.0.2.15	65.61.137.117	HTTP	404	GET /favicon.ico HTTP/1.1 (application/x-www-form-urlencoded)
802	50.709423	65.61.137.117	10.0.2.15	HTTP	306	HTTP/1.1 302 Found
804	50.713209	10.0.2.15	65.61.137.117	HTTP	597	GET /bank/main.jsp HTTP/1.1
808	50.881267	65.61.137.117	10.0.2.15	HTTP	3622	HTTP/1.1 200 OK (text/html)
838	60.340730	10.0.2.15	34.107.221.82	HTTP	342	GET /success.txt HTTP/1.1
830	60.376614	34.107.221.82	10.0.2.15	HTTP	270	HTTP/1.1 200 OK (text/plain)

File: "home/analyst/httpdump.pcap" ... Packets: 974 · Displayed: 39 (4.0%) · Load time: 0:00:04 · Profile: Default

- ▶ Frame 798: 589 bytes on wire (4712 bits), 589 bytes captured (4712 bits)
- ▶ Ethernet II, Src: PcsCompu_ab:b4:58 (08:00:27:ab:b4:58), Dst: 52:55:0a:00:02:02 (52:55:0a:00:02:02)
- ▶ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 65.61.137.117
- ▶ Transmission Control Protocol, Src Port: 33258, 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"

Parte 2: Cattura e visualizza il traffico HTTPS

In questa parte dell'esercizio, è stato utilizzato **tcpdump** per catturare il traffico HTTPS generato da una workstation Linux durante la navigazione su un sito web sicuro (www.netacad.com).

Il processo è iniziato con l'esecuzione di **tcpdump** con il comando **sudo tcpdump -i enp0s3 -s 0 -w httpsdump.pcap**, che ha avviato la cattura del traffico sulla specifica interfaccia di rete **enp0s3** e ha salvato l'output in un file denominato **httpsdump.pcap**. Durante la cattura, è stato aperto un browser web e il sito www.netacad.com è stato visitato. È stato notato che l'URL del sito utilizzava il protocollo HTTPS, il che implica l'uso di una connessione sicura (indicato dal lucchetto accanto all'URL).

Successivamente, è stato interrotto il comando **tcpdump** con **CTRL+C**, fermando così la registrazione del traffico di rete.

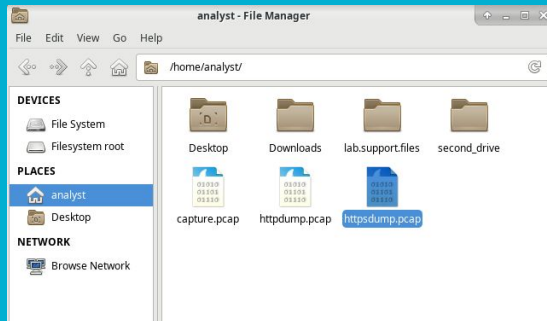
Per l'analisi, il file **httpsdump.pcap** è stato aperto in **Wireshark**. Utilizzando il filtro **tcp.port==443**, è stato possibile visualizzare solo il traffico relativo alla porta 443, che è tipica per il traffico HTTPS. Al contrario del traffico HTTP, che appare in chiaro, il traffico HTTPS è stato cifrato. Dopo la sezione TCP, la finestra di acquisizione ha mostrato una sezione **SSL/TLS 1.2**, indicante che il traffico è stato crittografato utilizzando il protocollo **TLS** (Transport Layer Security).

Espandendo completamente la sezione SSL/TLS, è stato possibile osservare che i **dati dell'applicazione** erano **crittografati**, rendendoli illeggibili. In sostanza, a differenza del traffico HTTP che trasmette dati in chiaro, il traffico HTTPS garantisce che il payload dei dati non sia visibile e che qualsiasi informazione sensibile, come credenziali di accesso o dati personali, sia protetta tramite crittografia. Questo rende HTTPS molto più sicuro rispetto a HTTP, poiché impedisce agli intrusi di intercettare o manipolare i dati durante la trasmissione.

In conclusione, l'esercizio ha evidenziato la differenza tra il traffico HTTP non sicuro e HTTPS, dimostrando come la crittografia SSL/TLS protegga i dati nelle comunicazioni online, rendendo impossibile la visualizzazione del contenuto dei pacchetti catturati con strumenti come Wireshark.

Parte 2: Cattura e visualizza il traffico HTTPS

```
Terminal - analyst@secOps:~  
File Edit View Terminal Tabs Help  
[analyst@secOps ~]$ sudo tcpdump -i enp0s3 -s 0 -w httpsdump.pcap  
[sudo] password for analyst:  
tcpdump: listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes  
^C987 packets captured  
987 packets received by filter  
0 packets dropped by kernel  
[analyst@secOps ~]$
```



Filter: tcp.port==443						Expression...	Clear	Apply	Save
No.	Time	Source	Destination	Protocol	Length	Info			
74	0.684203	34.120.5.221	10.0.2.15	TCP	60	443 → 57738 [ACK] Seq=3375 Ack=522 Win=65535 Len=0			
75	0.702034	34.120.5.221	10.0.2.15	TLSv1.2	92	Application Data			
76	0.702042	10.0.2.15	34.120.5.221	TCP	54	57740 → 443 [ACK] Seq=899 Ack=3393 Win=37440 Len=0			
77	0.707906	34.120.5.221	10.0.2.15	TLSv1.2	5918	Application Data, Application Data, Application Data, Application Data			
78	0.707916	10.0.2.15	34.120.5.221	TCP	54	57740 → 443 [ACK] Seq=899 Ack=3393 Win=37440 Len=0			

```
Frame 75: 92 bytes on wire (736 bits), 92 bytes captured (736 bits) on interface  
Ethernet II, Src: 52:55:0a:00:02:02 (52:55:0a:00:02:02), Dst: PcsCompu_ab:b4:58 (08:00:27:ab:b4:58)  
Internet Protocol Version 4, Src: 34.120.5.221, Dst: 10.0.2.15  
Transmission Control Protocol, Src Port: 443, Dst Port: 57740, Seq: 3375, Ack: 899, Len: 38  
Secure Sockets Layer  
TLSv1.2 Record Layer: Application Data Protocol: http2  
Content Type: Application Data (23)  
Version: TLS 1.2 (0x0303)  
Length: 33  
Encrypted Application Data: 0000000000000002df8b7f5a00254defa0f9322b22793f5...
```

Conclusioni

L'utilizzo di HTTPS offre numerosi vantaggi rispetto a HTTP, tra cui la protezione dei dati trasmessi tramite crittografia, che impedisce a terzi di intercettare o modificare le informazioni durante la trasmissione. HTTPS garantisce anche l'autenticità del sito web, riducendo il rischio di attacchi come il "man-in-the-middle". Inoltre, HTTPS è essenziale per proteggere le credenziali degli utenti e altre informazioni sensibili.

Tuttavia, non tutti i siti web che utilizzano HTTPS sono necessariamente affidabili. Anche se la connessione è cifrata, la presenza di un certificato SSL/TLS valido non implica automaticamente che il sito sia sicuro o che non sia stato compromesso. Un sito potrebbe utilizzare HTTPS ma comunque essere soggetto a vulnerabilità o contenere contenuti dannosi. Pertanto, è sempre importante valutare la reputazione del sito e la sicurezza complessiva.

Indice

Bonus:

- Laboratorio - Esplorazione di Nmap
- Attacco a un Database MySQL

Laboratorio - Esplorazione di Nmap

Obiettivi

- **Parte 1: Esplorazione di Nmap**
- **Parte 2: Scansione delle porte aperte**

Laboratorio - Esplorazione di Nmap

Nmap, abbreviazione di **Network Mapper**, è uno strumento open source ampiamente utilizzato per l'esplorazione delle reti e la scansione delle porte. Il suo scopo principale è fornire agli amministratori di sistema e agli analisti di sicurezza informazioni dettagliate sugli host presenti in una rete. Attraverso Nmap, è possibile identificare i dispositivi attivi, determinare quali porte sono aperte e rilevare i servizi in esecuzione su ciascun host. Inoltre, lo strumento consente di identificare i sistemi operativi utilizzati e di individuare eventuali vulnerabilità o configurazioni errate, rendendolo essenziale per gli audit di sicurezza. Nmap si rivela utile anche per la gestione delle reti aziendali, poiché consente di creare mappe dettagliate della rete e analizzare il comportamento dei servizi disponibili.

Parte 1: Esplorazione di Nmap

Navigazione nelle pagine di manuale

Durante la lettura del manuale Nmap (`man nmap`):

- Puoi utilizzare i tasti freccia su e giù per scorrere il documento.
- Premendo la barra spaziatrice, è possibile avanzare rapidamente di una pagina.
- La barra (/) consente di cercare termini o frasi in avanti. Ad esempio, digitando `/scan`, troverai informazioni correlate alla scansione.
- Il punto interrogativo (?) consente di effettuare ricerche all'indietro nel documento.
- Il tasto n permette di passare alla prossima occorrenza del termine cercato.

Utilizzo pratico di Nmap

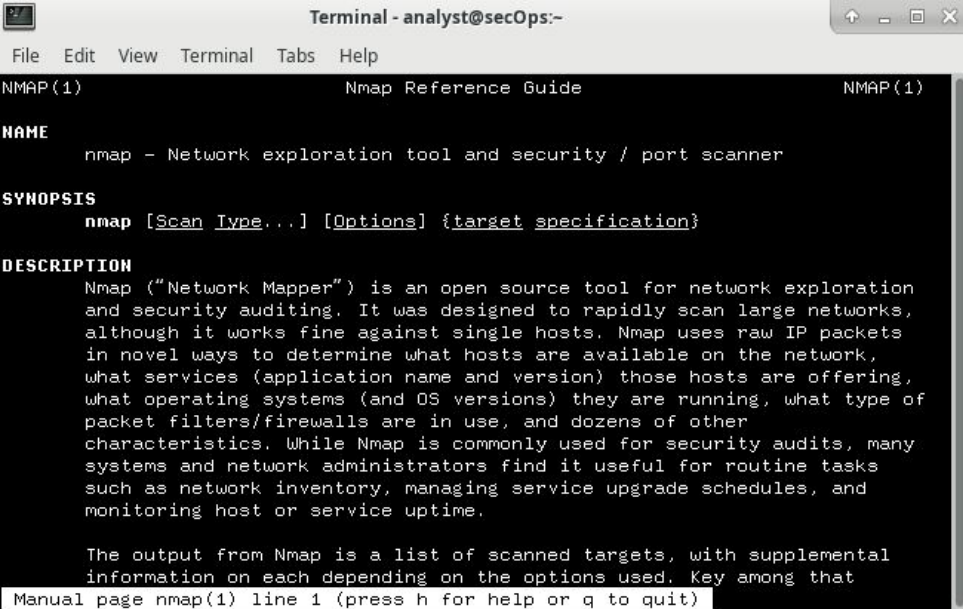
Nmap è uno strumento versatile e ampiamente utilizzato da amministratori di rete e specialisti di sicurezza. Consente di identificare:

- Host non autorizzati in una rete.
- Servizi che potrebbero esporre vulnerabilità.
- Porte aperte che necessitano di essere chiuse per migliorare la sicurezza.

Parte 1: Esplorazione di Nmap

Conclusione

Le pagine del manuale di Nmap sono una risorsa fondamentale per comprendere il suo utilizzo, i parametri e i comandi disponibili. Grazie alla loro struttura intuitiva, è possibile approfondire le funzionalità di questo strumento, che è cruciale per l'analisi di sicurezza e il monitoraggio delle reti.



```
Terminal - analyst@secOps:~
File Edit View Terminal Tabs Help
NMAP(1) Nmap Reference Guide NMAP(1)

NAME
    nmap - Network exploration tool and security / port scanner

SYNOPSIS
    nmap [Scan Type...] [Options] {target specification}

DESCRIPTION
    Nmap ("Network Mapper") is an open source tool for network exploration
    and security auditing. It was designed to rapidly scan large networks,
    although it works fine against single hosts. Nmap uses raw IP packets
    in novel ways to determine what hosts are available on the network,
    what services (application name and version) those hosts are offering,
    what operating systems (and OS versions) they are running, what type of
    packet filters/firewalls are in use, and dozens of other
    characteristics. While Nmap is commonly used for security audits, many
    systems and network administrators find it useful for routine tasks
    such as network inventory, managing service upgrade schedules, and
    monitoring host or service uptime.

    The output from Nmap is a list of scanned targets, with supplemental
    information on each depending on the options used. Key among that

Manual page nmap(1) line 1 (press h for help or q to quit)
```

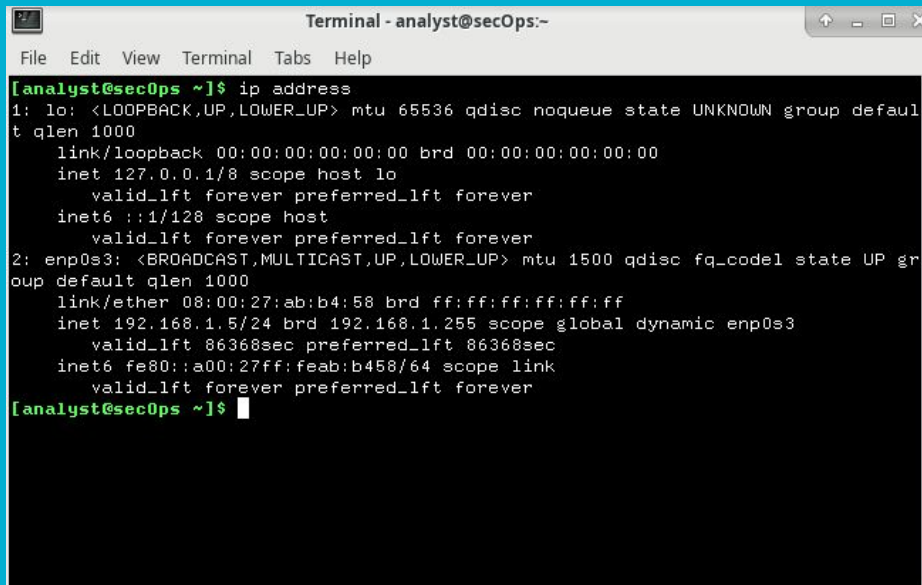
Parte 2: Scansione delle porte aperte

La VM appartiene alla rete **192.168.1.0/24**, con un indirizzo IP assegnato di **192.168.1.5/24**, determinato attraverso il comando `ip address`.

Successivamente, è stata eseguita una scansione sul server remoto **scanme.nmap.org** utilizzando il comando `nmap -A -T4 scanme.nmap.org`. Questo sito è progettato per consentire agli utenti di apprendere come utilizzare Nmap e testare la propria installazione.

Risultati della scansione:

1. **Porte e servizi aperti:**
 - **22/tcp:** SSH
 - **9929/tcp:** n ping-echo
 - **31337/tcp:** tcpwrapped
 - **80/tcp:** HTTP
2. **Indirizzo IP del server:**
 - **IPv4:** 45.33.32.156
 - **IPv6:** 2600:3c01::f03c:91ff:fe18:bb2f
3. **Sistema operativo:** Ubuntu Linux



```
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
    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:ab:b4:58 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.5/24 brd 192.168.1.255 scope global dynamic enp0s3
        valid_lft 86368sec preferred_lft 86368sec
    inet6 fe80::a00:27ff:feab:b458/64 scope link
        valid_lft forever preferred_lft forever
[analyst@secOps ~]$
```

Parte 2: Scansione delle porte aperte

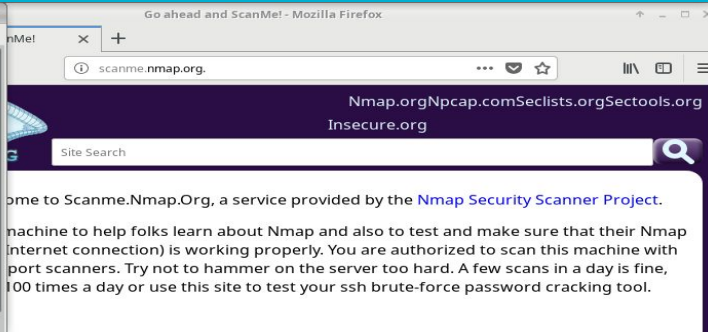
Questa scansione ha confermato che il server remoto utilizza Ubuntu Linux e offre servizi chiave come SSH e HTTP. Le informazioni raccolte sono utili per l'analisi delle vulnerabilità e per comprendere la configurazione del server remoto.

```
analyst@secOps ~]$ nmap -A -T4 localhost
Starting Nmap 7.70 ( https://nmap.org ) at 2024-12-13 07:33 EST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0000s latency).
Other addresses for localhost (not scanned): ::1
Not shown: 996 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
22/tcp    open  vsftpd 2.0.8 or later
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_vsftpd-__ 1 0 0 0 Mar 26 2019 ftp_test
|_ftp-__ 1 0 0 0 Mar 26 2019 ftp_test
|_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:d5:79:25:06:44:c7:9e:f0:e0:e7:5b:bb (RSA)
|_ 256 06:12:78:fe:b3:89:29:4f:8d:f3:9e:9e:d7:d6:03:52 (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 11.47 seconds
analyst@secOps ~]$
```

```
analyst@secOps ~]$ nmap -A -T4 scanme.nmap.org
Starting Nmap 7.70 ( https://nmap.org ) at 2024-12-13 09:58 EST
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.19s latency).
Other addresses for scanme.nmap.org (not scanned): 2600:3c01::f03c:91ff:fe18:bb2
Not shown: 996 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
22/tcp    open  OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.13 (Ubuntu Linux; prod)
|_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:b6:b3:05:14:c2:a6:b2 (RSA)
|_ 256 96:02:bb:5e:57:54:1c:4e:4b:2f:56:4c:4a:24:b2:57 (ECDSA)
|_ 256 33:fa:91:0f:e0:e1:7b:1f:6d:05:a2:00:f1:54:41:56 (ED25519)
80/tcp    open  http
|_http-server-header: Apache/2.4.7 (Ubuntu)
|_http-title: Go ahead and ScanMe!
929/tcp   open  ping-sctp Nping echo
81337/tcp open  cspwrapd
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
```



Attacco a un Database MySQL

Obiettivi

- **Parte 1: aprire Wireshark e caricare il file PCAP.**
- **Parte 2: Visualizza l'attacco SQL Injection.**
- **Parte 3: L'attacco SQL Injection continua...**
- **Parte 4: L'attacco SQL Injection fornisce informazioni di sistema.**
- **Parte 5: L'attacco SQL Injection e le informazioni della tabella**
- **Parte 6: Conclusione dell'attacco SQL Injection.**

Parte 1: aprire Wireshark e caricare il file PCAP.

Il file **SQL_Lab.pcap** contiene una cattura di pacchetti di rete che documenta un attacco di iniezione SQL. Analizzando il traffico, possiamo identificare i due indirizzi IP coinvolti nell'attacco di iniezione SQL.

In Wireshark, i pacchetti mostrano che:

- **Indirizzo IP 10.0.2.4** è l'indirizzo dell'attaccante, che invia le richieste di iniezione SQL.
- **Indirizzo IP 10.0.2.15** è l'indirizzo del server vulnerabile, che riceve le richieste di iniezione SQL.

Durante l'attacco, l'attaccante (10.0.2.4) sfrutta una vulnerabilità nel sistema per inviare comandi SQL malformati al server (10.0.2.15). Questo tipo di attacco può compromettere la sicurezza del server, permettendo all'attaccante di manipolare il database, ottenere informazioni sensibili o eseguire operazioni non autorizzate.

In sintesi, gli indirizzi IP coinvolti sono:

- **10.0.2.4**: l'attaccante
- **10.0.2.15**: il server vulnerabile

No.	Time	Source	Destination	Protocol	Length	Info
16	220.490531	10.0.2.4	10.0.2.15	HTTP	577	GET /dwa/vulnerabilities/sql?id=1%27+or+%270%27%3D%270+&Submit=Submit HTTP/1.1
17	220.490637	10.0.2.15	10.0.2.4	TCP	66	80 → 35640 [ACK] Seq=1 Ack=512 Win=235 Len=0 TSval=93660 TSecr=111985
18	220.493085	10.0.2.15	10.0.2.4	HTTP	1918	HTTP/1.1 200 OK (text/html)
19	277.727722	10.0.2.4	10.0.2.15	HTTP	630	GET /dwa/vulnerabilities/sql?id=1%27+or+1%3D1+union+select+database%28%29%2C+user%28%29%238.Submit=
20	277.727871	10.0.2.15	10.0.2.4	TCP	66	80 → 35642 [ACK] Seq=1 Ack=565 Win=236 Len=0 TSval=107970 TSecr=129156
21	277.732200	10.0.2.15	10.0.2.4	HTTP	1955	HTTP/1.1 200 OK (text/html)
22	313.710129	10.0.2.4	10.0.2.15	HTTP	659	GET /dwa/vulnerabilities/sql?id=1%27+or+1%3D1+union+select+null%2C+version+%28%29%238.Submit=Submit HT
23	313.710277	10.0.2.15	10.0.2.4	TCP	66	80 → 35644 [ACK] Seq=1 Ack=594 Win=236 Len=0 TSval=116966 TSecr=139951
24	313.712414	10.0.2.15	10.0.2.4	HTTP	1954	HTTP/1.1 200 OK (text/html)
25	383.277032	10.0.2.4	10.0.2.15	HTTP	680	GET /dwa/vulnerabilities/sql?id=1%27+or+1%3D1+union+select+null%2C+table_name+from+information_schemata
26	383.277811	10.0.2.15	10.0.2.4	TCP	66	80 → 35666 [ACK] Seq=1 Ack=615 Win=236 Len=0 TSval=134358 TSecr=160821
27	383.284289	10.0.2.15	10.0.2.4	HTTP	4068	HTTP/1.1 200 OK (text/html)
28	441.804070	10.0.2.4	10.0.2.15	HTTP	685	GET /dwa/vulnerabilities/sql?id=1%27+or+1%3D1+union+select+user%2C+password+from+users%238.Submit=Sub
29	441.804427	10.0.2.15	10.0.2.4	TCP	66	80 → 35668 [ACK] Seq=1 Ack=620 Win=236 Len=0 TSval=148990 TSecr=178379
30	441.807206	10.0.2.15	10.0.2.4	HTTP	2091	HTTP/1.1 200 OK (text/html)

Parte 2: Visualizza l'attacco SQL Injection.

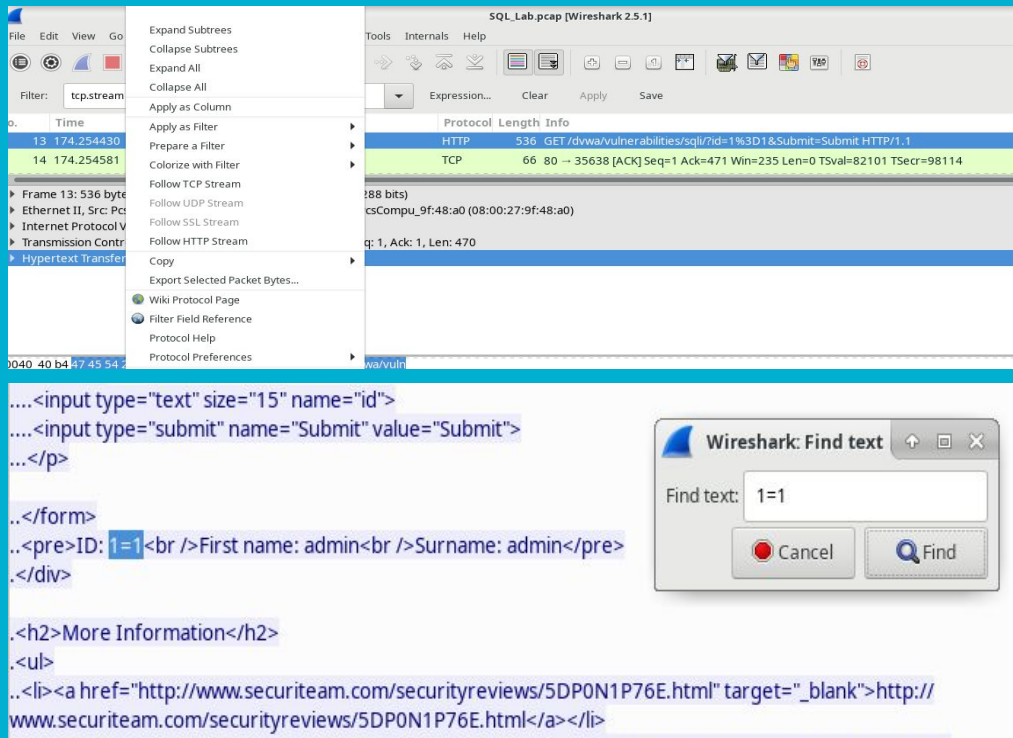
In questa fase, l'attacco di **SQL Injection** diventa più chiaro, poiché si osserva l'interazione tra l'attaccante e il server vulnerabile tramite una richiesta HTTP GET.

Ecco una sintesi dei passaggi descritti:

1. **Flusso HTTP:** L'attaccante invia una richiesta GET al server (10.0.2.15) sulla riga 13 della cattura di Wireshark. Questa richiesta è un tentativo di sfruttare una vulnerabilità nel sito web target. Il traffico di rete proveniente dall'attaccante appare in **rosso**, mentre la risposta del server è mostrata in **blu**.
2. **Ricerca "1=1":** Nel campo di ricerca della finestra di **Follow HTTP Stream**, l'attaccante inserisce la stringa **"1=1"**. Questo è un classico esempio di un tentativo di SQL Injection. La query SQL **"1=1"** è sempre vera e viene utilizzata per testare se il server è vulnerabile a iniezioni SQL. Se il sistema non restituisce un messaggio di errore, ma invece fornisce dati dal database, significa che l'applicazione è vulnerabile.
3. **Verifica dell'iniezione:** L'inserimento di **"1=1"** nel campo UserID del sito ha permesso all'attaccante di vedere che il server risponde con un record del database. Questo comportamento dimostra che l'iniezione SQL è riuscita, in quanto la query **"1=1"** ha aggirato i controlli di accesso del sistema e ha restituito informazioni sensibili dal database, confermando la vulnerabilità.
4. **Chiusura della finestra di flusso:** Dopo aver verificato l'iniezione SQL, l'attaccante chiude la finestra di flusso HTTP per tornare alla visualizzazione dell'intera cattura in Wireshark.

Parte 2: Visualizza l'attacco SQL Injection.

In conclusione, l'attaccante ha utilizzato la tecnica di iniezione SQL per verificare la vulnerabilità del server. L'inserimento della query "1=1" ha portato a una risposta positiva dal server, rivelando che il sistema è suscettibile a un'iniezione SQL, che può essere ulteriormente sfruttata per ottenere accesso ai dati del database.



Parte 3: L'attacco SQL Injection continua

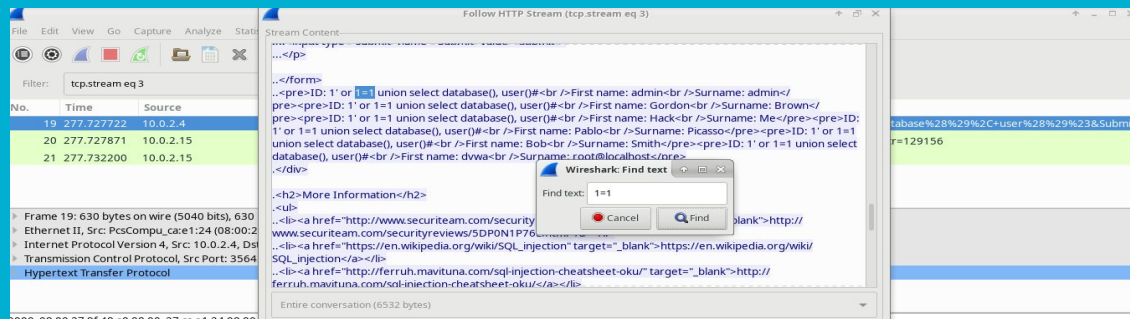
In questa fase del laboratorio, si osserva come l'attacco di **SQL Injection** continui a evolversi, portando l'attaccante a raccogliere ulteriori informazioni sensibili dal database.

Ecco una sintesi dei passaggi descritti:

1. **Flusso HTTP (Riga 19):** Dopo aver selezionato la riga 19 in Wireshark, l'attaccante continua a seguire il flusso HTTP per monitorare la conversazione tra il client e il server. Questo passaggio consente di visualizzare il traffico più dettagliatamente e vedere le risposte successive del server.
2. **Ricerca "1=1":** L'attaccante inserisce nuovamente la stringa **"1=1"** nel campo di ricerca per identificare se il server è vulnerabile, come già visto nella parte precedente dell'attacco.
3. **SQL Injection avanzata:** Successivamente, l'attaccante utilizza una query più complessa: **"1' or 1=1 union select database(), user()#"**. Questa query tenta di:
 - **1' or 1=1:** Chiudere prematuramente l'input dell'utente e forzare la condizione SQL a diventare sempre vera.
 - **union select database(), user():** Utilizzare la funzione `union select` per ottenere informazioni aggiuntive, come il nome del database e l'utente che sta eseguendo la query.
 - **#:** Commentare il resto della query, ignorando eventuali errori.
4. **Risposta del server:** Il server risponde con informazioni sensibili, tra cui:
 - Il nome del database: **dvwa**.
 - L'utente del database: **root@localhost**.
 - Vengono anche restituiti dettagli sugli account utente del database, confermando che l'attaccante ha ottenuto informazioni preziose sul sistema.
5. **Chiusura e visualizzazione completa:** Dopo aver esaminato la risposta del server, l'attaccante chiude la finestra del flusso HTTP e torna a visualizzare l'intera cattura di pacchetti in Wireshark, per analizzare altre possibili vulnerabilità o tracce dell'attacco.

Parte 3: L'attacco SQL Injection continua

In sintesi, l'attacco SQL Injection continua con un tentativo di ottenere informazioni critiche dal database, come il nome del database (**dvwa**) e l'utente del database (**root@localhost**). La risposta positiva indica che l'attaccante ha riuscito a manipolare la query SQL e ad accedere a informazioni sensibili, il che evidenzia la vulnerabilità del sistema a questo tipo di attacco.



```
..</form>
..<pre>ID: 1' or 1=1 union select database(), user()#<br />First name: admin<br />Surname: admin</pre>
<pre><pre>ID: 1' or 1=1 union select database(), user()#<br />First name: Gordon<br />Surname: Brown</pre>
<pre><pre>ID: 1' or 1=1 union select database(), user()#<br />First name: Hack<br />Surname: Me</pre>
<pre><pre>ID: 1' or 1=1 union select database(), user()#<br />First name: Pablo<br />Surname: Picasso</pre>
<pre><pre>ID: 1' or 1=1 union select database(), user()#<br />First name: Bob<br />Surname: Smith</pre>
<pre><pre>ID: 1' or 1=1 union select database(), user()#<br />First name: dvwa<br />Surname: root@localhost</pre>
..</div>

..<h2>More Information</h2>
..<ul>
..<li><a href="http://www.securiteam.com/securityreviews/5DP0N1P76E.html">http://www.securiteam.com/securityreviews/5DP0N1P76E.html</a>
..<li><a href="https://en.wikipedia.org/wiki/SQL_injection" target="_blank">https://en.wikipedia.org/wiki/
```

Parte 4: L'attacco SQL Injection fornisce informazioni di sistema.

In questa fase, l'attacco SQL Injection avanza ulteriormente, con l'aggressore che mira a ottenere informazioni specifiche sul sistema, in particolare la **versione del database MySQL**.

Ecco una sintesi dei passaggi descritti:

1. **Flusso HTTP (Riga 22):** L'attaccante seleziona la riga 22 in Wireshark per seguire il flusso HTTP tra il client e il server. In questo flusso, il traffico sorgente appare in **rosso** (richiesta GET inviata dal client), mentre la risposta del server è in **blu** (dati restituiti dal server).
2. **Ricerca "1=1":** Come nelle fasi precedenti, l'attaccante cerca la stringa "1=1" per verificare se il server è ancora vulnerabile all'iniezione SQL.
3. **Query per identificare la versione:** L'attaccante inserisce una query avanzata per ottenere la versione del sistema MySQL:
 - **"1' or 1=1 union select null, version()#"**.
 - **1' or 1=1:** Chiude l'input dell'utente e forza la condizione SQL a essere sempre vera.
 - **union select null, version():** Utilizza il comando union select per ottenere la versione di MySQL.
 - **#:** Commenta il resto della query per evitare errori.
4. **Risposta del server:** La risposta del server restituisce l'informazione desiderata: la versione di MySQL. Alla fine dell'output, prima del codice HTML di chiusura `</pre></div>`, l'attaccante trova la stringa **"Versione MySQL 5.7.12-0"**, che indica che il server sta utilizzando una versione specifica di MySQL.
5. **Chiusura e visualizzazione completa:** Dopo aver ottenuto l'informazione sulla versione di MySQL, l'attaccante chiude la finestra del flusso HTTP e torna a visualizzare l'intera cattura di pacchetti in Wireshark per analizzare ulteriori dettagli dell'attacco.

Parte 4: L'attacco SQL Injection fornisce informazioni di sistema.

In sintesi, l'attacco SQL Injection è stato utilizzato per ottenere informazioni precise sulla versione del database MySQL, in particolare la versione **5.7.12-0**. Questa informazione è cruciale per l'aggressore, poiché consente di conoscere eventuali vulnerabilità specifiche della versione di MySQL utilizzata, e può essere sfruttata per lanciare attacchi mirati.

The top screenshot shows the Wireshark interface with the 'Follow HTTP Stream' window open. The 'Stream Content' pane displays the HTML response from the server, which includes a list of links to security resources. The 'Find text' dialog box is open, showing the search results for the injected payload.

The bottom screenshot shows the 'Find text' dialog box with the search results for the injected payload. The search results show the following text:

```
..</form>
..<pre>ID: 1' or 1=1 union select null, version ()#<br />First name: admin<br />Surname: admin</pre><pre>ID: 1'
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>
..<h2>More Information</h2>
..<ul>
..<li><a href="http://www.securiteam.com/securityreviews/SDPON1P76E.H" target="_blank">http://
www.securiteam.com/securityreviews/SDPON1P76E.H
..<li><a href="https://en.wikipedia.org/wiki/SQL_injection" target="_blank">https://en.wikipedia.org/wiki/
SQL_injection</a></li>
..<li><a href="http://ferruh.mavituna.com/sqli-injection-cheatsheet-oku/" target="_blank">http://
ferruh.mavituna.com/sqli-injection-cheatsheet-oku/</a></li>
..<li><a href="http://bentestmonkev.net/cheat-sheet/sqli-injection/mssql-sqli-injection-cheat-sheet" target="_blank">http://
bentestmonkev.net/cheat-sheet/sqli-injection/mssql-sqli-injection-cheat-sheet</a></li>
..</li>
</ul>
</div>
```


Parte 5: L'attacco SQL Injection e le informazioni della tabella

In questa fase, l'attaccante cerca di raccogliere informazioni più dettagliate sulle **tabelle del database** target, utilizzando tecniche di **SQL Injection** per esplorare la struttura del database e identificare tabelle specifiche contenenti informazioni sensibili.

Ecco una sintesi dei passaggi descritti:

1. **Flusso HTTP (Riga 25):** L'attaccante seleziona la riga 25 in Wireshark per seguire il flusso HTTP tra il client (attaccante) e il server. Il traffico sorgente appare in **rosso** (richiesta GET inviata dall'attaccante), mentre la risposta del server è in **blu** (dati restituiti dal server).
2. **Ricerca "utenti":** L'attaccante inserisce la parola "**utenti**" nel campo di ricerca in Wireshark, per individuare riferimenti a tabelle o colonne che potrebbero contenere informazioni sugli utenti.
 - L'attaccante modifica la query per includere una ricerca più specifica: **"1' OR 1=1 UNION SELECT null, column_name FROM INFORMATION_SCHEMA.columns WHERE table_name='users'"**.
 - **Cosa fa questa query?:** La query cerca nel database le colonne della tabella denominata "**users**" (un nome tipico di una tabella contenente informazioni sugli utenti). La query utilizza `INFORMATION_SCHEMA.columns` per ottenere i nomi delle colonne della tabella `users`. Se il database contiene una tabella con questo nome, l'attaccante otterrà i nomi delle colonne associate a questa tabella.
 - Il database restituirà un output **più breve** filtrato dalla parola "`users`", permettendo all'attaccante di identificare rapidamente la struttura della tabella.

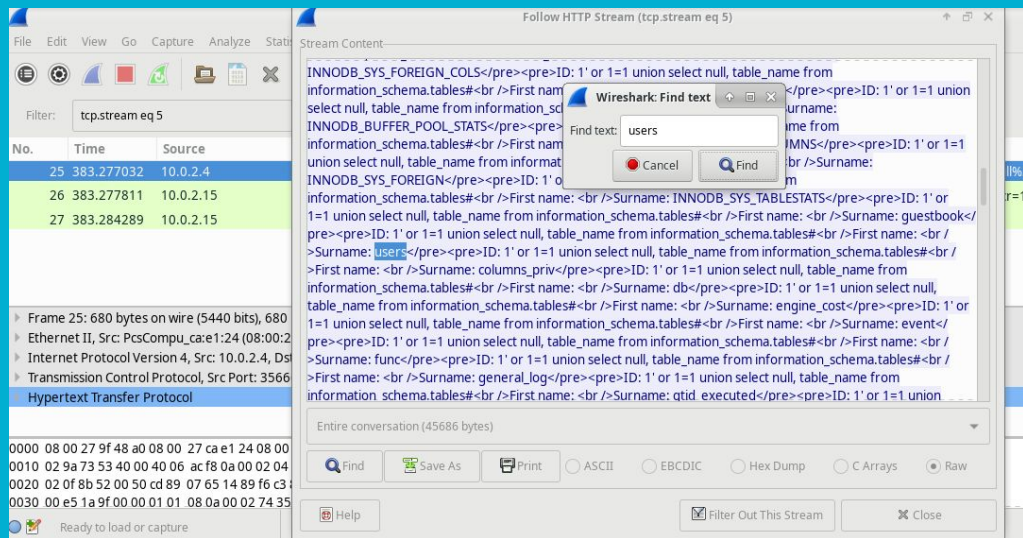
Parte 5: L'attacco SQL Injection e le informazioni della tabella

Visualizzazione delle tabelle (Query avanzata):

Successivamente, l'attaccante inserisce una query per ottenere tutte le tabelle presenti nel database:

- **"1' OR 1=1 UNION SELECT null, table_name FROM information_schema.tables#"**.
- Questa query cerca di ottenere il nome di tutte le tabelle nel database utilizzando INFORMATION_SCHEMA.tables. La risposta restituirà **molte tabelle**, poiché l'attaccante ha utilizzato **"null"** come primo valore nel SELECT, senza specificare un altro valore utile, il che porta a un output ampio di molte tabelle presenti nel database.

Chiusura e visualizzazione completa: Dopo aver esaminato i risultati delle tabelle, l'attaccante chiude la finestra di flusso HTTP e torna a visualizzare l'intera conversazione in Wireshark, per analizzare altre informazioni utili sull'attacco e su eventuali altre vulnerabilità.



Parte 6: Conclusione dell'attacco SQL Injection.

In questa fase finale dell'attacco di **SQL Injection**, l'attaccante è riuscito a ottenere uno degli obiettivi principali: **gli hash delle password** degli utenti nel database.

Ecco una sintesi dei passaggi:

1. **Flusso HTTP (Riga 28):** L'attaccante seleziona la riga 28 in Wireshark per seguire il flusso HTTP. La richiesta GET inviata dal client appare in **rosso**, mentre la risposta del server è in **blu**.
2. **Ricerca "1=1":** L'attaccante cerca la stringa **"1=1"** nel flusso HTTP, per individuare la parte in cui viene inviata una richiesta di iniezione SQL.
3. **Query per ottenere utenti e hash delle password:** L'attaccante inserisce la query **"1' OR 1=1 UNION SELECT user, password FROM users#"**. Questa query tenta di estrarre i **nomi utente** e **gli hash delle password** dalla tabella `users`. In questo caso, il server risponde con l'hash della password per l'utente **1337**, il cui hash è **"8d3533d75ae2c3966d7e0d4fcc69216b"**.
4. **Decodifica dell'hash:** L'attaccante utilizza uno strumento online, come **CrackStation** (<https://crackstation.net/>), per craccare l'hash. Dopo aver inserito l'hash **"8d3533d75ae2c3966d7e0d4fcc69216b"**, lo strumento restituisce la **password in chiaro: "Carlo"**.
5. **Chiusura e conclusione:** Dopo aver ottenuto la password in chiaro, l'attaccante chiude la finestra di flusso HTTP e tutte le altre finestre aperte in Wireshark, completando così l'attacco.

