Cyber Security & Ethical Hacking Progetto

profilo studente

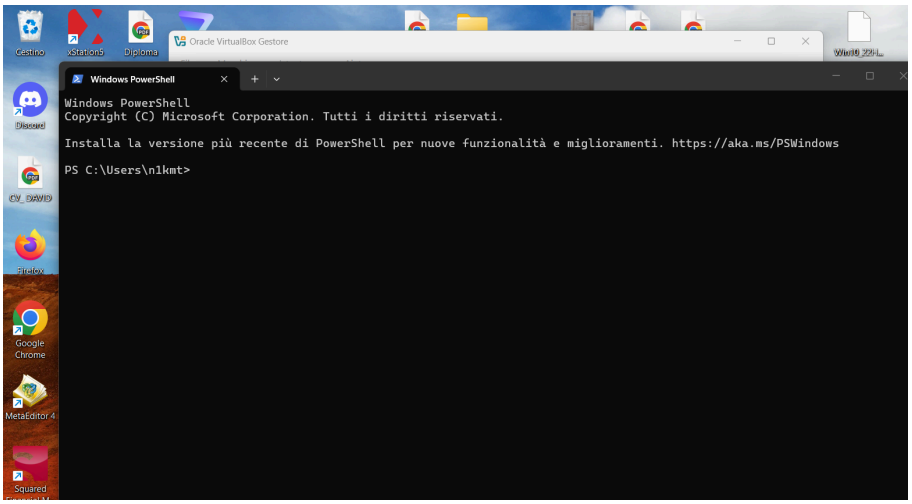
- **Gabriel Giustinelli**
- **Epicode Cyber Security**
- **classe** CS0525
- **data** 12/02/2026

1. Windows PowerShell

1.1 OBIETTIVO

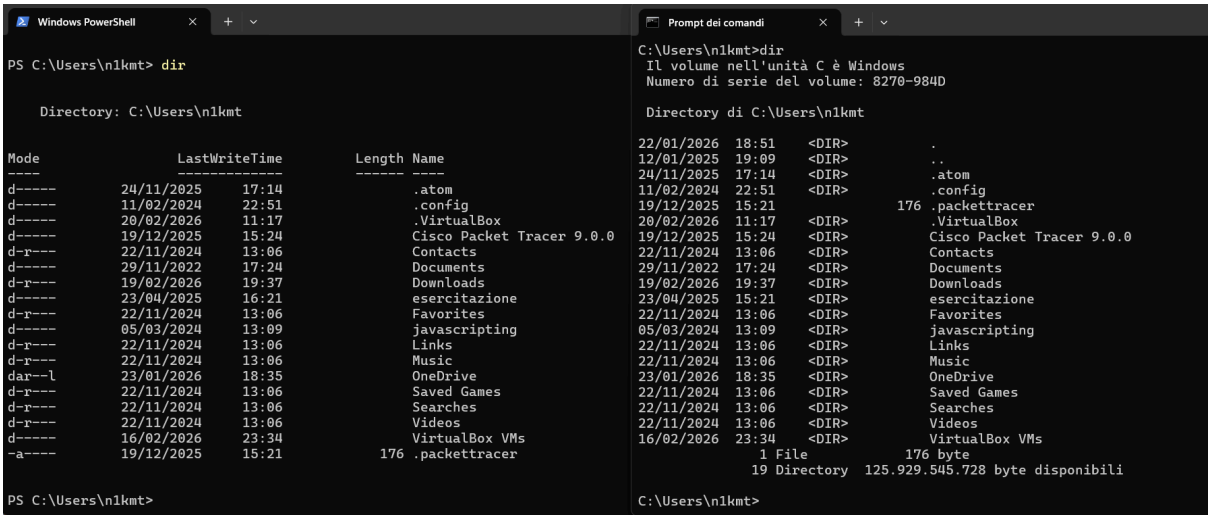
L'obiettivo di questa unità è dimostrare la padronanza della shell avanzata **Microsoft PowerShell** per l'amministrazione del sistema e l'analisi della sicurezza. L'attenzione è posta sulla capacità di:

- Interagire con il Sistema Operativo tramite cmdlet e oggetti.
- Identificare processi attivi e servizi di sistema.
- Monitorare le connessioni di rete per individuare potenziali attività anomale.



1.2 Analisi Comparativa: CMD vs PowerShell

Effettuiamo un confronto tra il Prompt dei Comandi **CMD** e **PowerShell** di Windows utilizzando il comando **dir**.



- **Risultato:** L'output appare simile, ma PowerShell visualizza più dettagli (come la modalità **`d----`** per le cartelle), in PowerShell **dir** è un "alias" del cmdlet **Get-ChildItem**
- **Nota:** A differenza del **CMD**, che restituisce solo testo, **PowerShell** lavora con **oggetti**. Questo permette all'analista di filtrare e manipolare i dati (file, directory, permessi).

Domanda: Quali sono gli output dei comandi `dir`?

- **CMD:** L'output è un elenco testuale semplice. Mostra la data, l'ora, se si tratta di una directory **`<DIR>`** o di un file (con la dimensione in byte) e il nome.
- **PowerShell:** L'output è più dettagliato e strutturato in colonne. Le colonne principali sono **Mode**, **LastWriteTime**, **Length** e **Name**.

Provando altri comandi classici come **ping**, **cd**, **ipconfig**. Sia nel CMD che in PowerShell possiamo esaminare i risultati.

Domanda: Quali sono i risultati?

I risultati sono identici a quelli che otterresti nel Prompt dei Comandi CMD

- **`ipconfig`**, PowerShell mostra tutti i dettagli della connessione di rete.
- **`ping google.it`**, il comando funziona normalmente e mostra i tempi di risposta.
- **`cd`**, ci sposta tra le cartelle senza problemi.

In sintesi: Tutti i comandi classici del **CMD** funzionano anche in **PowerShell** perché PowerShell è stato progettato per riconoscerli e "tradurli" automaticamente.

1.3 Identificare i comandi in PowerShell

In PowerShell tutti i comandi sono formati da due parole unite da un trattino. Seguono lo schema **Verbo-Sostantivo**.

- Il **Verbo** (es. **`Get`**) indica l'azione: "prendi" o "mostrami".
- Il **Sostantivo** (es. **`Service`** o **`Process`**) indica su cosa stiamo lavorando.

Per identificare il comando PowerShell ed elencare le sottodirectory e i file in una directory, inseriamo **Get-Alias dir** al prompt di PowerShell

```
PS C:\Users\n1kmt> Get-Alias dir

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


PS C:\Users\n1kmt>
```

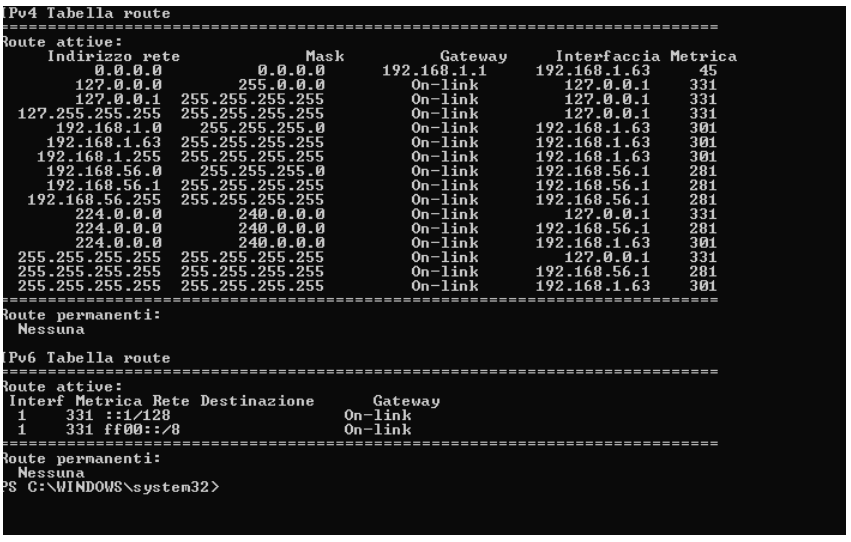
Domanda: Qual è il comando PowerShell per dir?

Quindi come abbiamo visto poco fa analizzando le differenze tra CMD e PowerShell, Il comando PowerShell "reale" per **dir** è: **Get-ChildItem**

1.4 Comando netstat in PowerShell

1.4. 1 Analisi del Routing e del Gateway Predefinito

Per identificare il punto di uscita della rete locale, è stata analizzata la tabella di routing del sistema tramite il comando: **netstat -r**



```
IPv4 Tabella route
===========================================================================
Route attive:
     Indirizzo rete            Mask          Gateway     Interfaccia Metrica
          0.0.0.0          0.0.0.0    192.168.1.1    192.168.1.63     45
        127.0.0.0        255.0.0.0         On-link         127.0.0.1    331
        127.0.0.1  255.255.255.255         On-link         127.0.0.1    331
  127.255.255.255  255.255.255.255         On-link         127.0.0.1    331
      192.168.1.0    255.255.255.0         On-link      192.168.1.63    301
     192.168.1.63  255.255.255.255         On-link      192.168.1.63    301
    192.168.1.255  255.255.255.255         On-link      192.168.1.63    301
     192.168.56.0    255.255.255.0         On-link      192.168.56.1    281
     192.168.56.1  255.255.255.255         On-link      192.168.56.1    281
   192.168.56.255  255.255.255.255         On-link      192.168.56.1    281
        224.0.0.0        240.0.0.0         On-link         127.0.0.1    331
        224.0.0.0        240.0.0.0         On-link      192.168.56.1    281
        224.0.0.0        240.0.0.0         On-link      192.168.1.63    301
  255.255.255.255  255.255.255.255         On-link         127.0.0.1    331
  255.255.255.255  255.255.255.255         On-link      192.168.56.1    281
  255.255.255.255  255.255.255.255         On-link      192.168.1.63    301
===========================================================================
Route permanenti:
  Nessuna

IPv6 Tabella route
===========================================================================
Route attive:
 Interf Metrica Rete Destinazione      Gateway
  1    331 ::1/128                  On-link
  1    331 ff00::/8                 On-link
===========================================================================
Route permanenti:
  Nessuna
PS C:\WINDOWS\system32>
```

Domanda: Qual è il gateway IPv4?

Risultato:

- **Gateway IPv4 identificato:** 192.168.1.1
- **Interfaccia:** 192.168.1.63

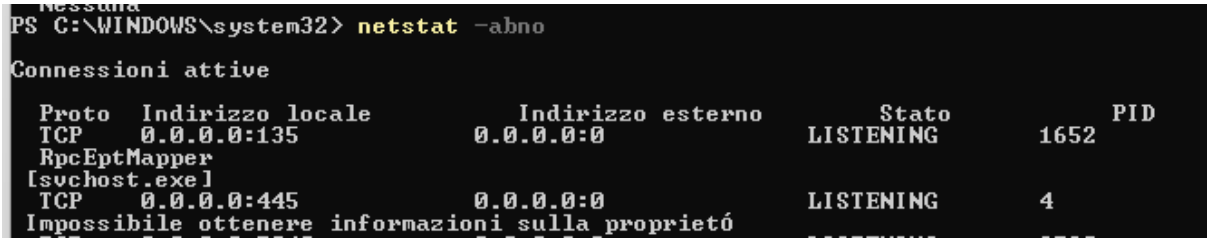
L'utilizzo del parametro **`-r`** ha permesso di esaminare la tabella di instradamento IP. infatti il monitoraggio della tabella di routing è fondamentale per rilevare eventuali **rotte statiche**

malevole inserite da software sospetti per deviare il traffico verso server esterni non autorizzati

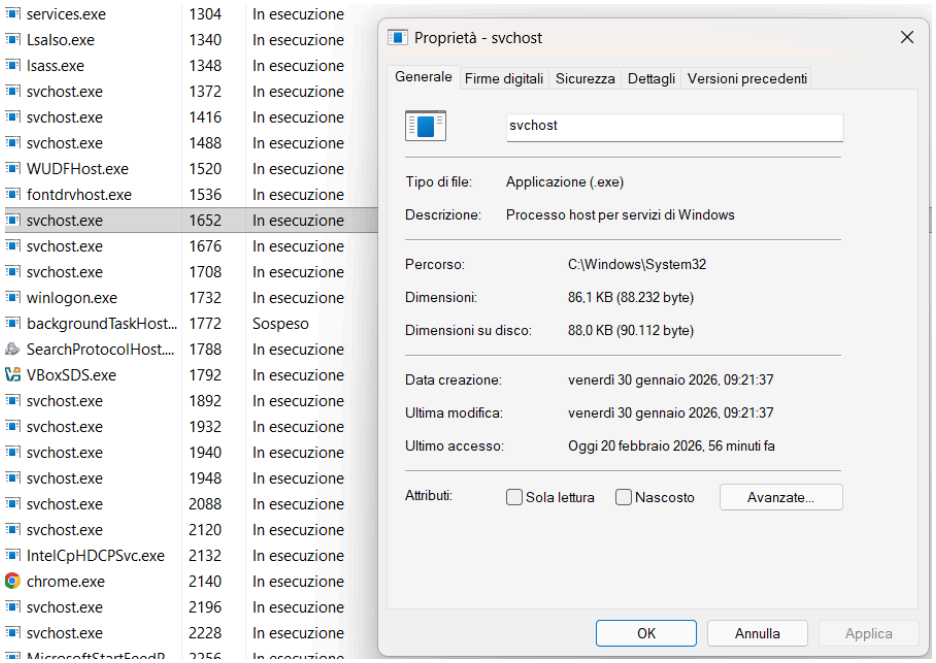
1.4. 2 Correlazione Processi e Connessioni di Rete

Dopo aver analizzato le connessioni attive tramite PowerShell, eseguiremo una verifica incrociata con gli strumenti di monitoraggio di sistema.

1. **Analisi PowerShell:** Utilizzando il comando **`netstat`**, è stato isolato il **PID 1652**.



2. **Verifica Task Manager:** All'interno della scheda **Dettagli**, il PID 1652 è stato identificato come il processo **svchost.exe**.



3. **Analisi Forense delle Proprietà:** L'analisi della finestra "Proprietà" (come mostrato nello screenshot) ha confermato che il processo risiede nel percorso legittimo **`C:\Windows\System32`**.

Domanda: Quali informazioni puoi ottenere dalla scheda Dettagli e dalla finestra di dialogo Proprietà per il PID selezionato?

Dalla scheda **Dettagli** e della finestra **Proprietà** per il PID **1652**, si possono ottenere i seguenti dati:

- **Identità del Processo:** Il nome dell'eseguibile **`svchost.exe`** e la sua descrizione specifica ("Processo host per servizi di Windows").
- **Percorso nel File System:** L'esatta posizione del file sul disco **`C:\Windows\System32`**, fondamentale per capire se si tratta di un file di sistema legittimo o di un malware camuffato.
- **Dettagli Temporali:** La data di creazione, l'ultima modifica e l'ultimo accesso al file.
- **Firme Digitali:** (Visibili nella scheda "Firme digitali" della finestra Proprietà) permettono di verificare l'autenticità del produttore (Microsoft in questo caso).
- **Dimensioni e Attributi:** Il peso del file su disco e se è impostato come "Sola lettura" o "Nascosto".

1.5 Gestione del File System e Automazione

Testiamo la capacità di PowerShell di interagire direttamente con i componenti del sistema operativo, in particolare per la gestione dello spazio disco.

Eseguendo il comando per lo svuotamento del Cestino di Windows: **Clear-RecycleBin**

Conferma di sicurezza: All'invio del comando, PowerShell non procede immediatamente ma richiede una conferma esplicita (**`Sì/No`**). Questo mostra una protezione contro l'esecuzione accidentale di comandi distruttivi.

Efficienza operativa: A differenza dell'interfaccia grafica, che richiede la navigazione sul desktop e l'interazione con il menu contestuale, il cmdlet permette di liberare risorse istantaneamente dal terminale.

Domanda: Cosa è successo ai file nel Cestino?

Dopo l'esecuzione del comando **`Clear-RecycleBin`**, i file che erano stati precedentemente eliminati sono stati **rimossi definitivamente** dal supporto di memorizzazione logico del sistema operativo.

- **A livello visivo:** L'icona del Cestino sul desktop appare ora vuota e non è più possibile ripristinare i file tramite l'interfaccia grafica.
- **A livello tecnico:** Il sistema ha liberato i puntatori ai file e lo spazio su disco che occupavano è stato contrassegnato come "disponibile" per nuove scritture.

Domanda: PowerShell è stato sviluppato per l'automazione delle attività. Quali comandi semplificano i compiti di un analista di sicurezza?

L'esercizio ha dimostrato che **PowerShell** non è solo un'alternativa al CMD, ma una **piattaforma di controllo completa**. Attraverso la ricerca, ho identificato tre aree chiave in cui PowerShell automatizza e semplifica il lavoro di un analista:

1. **Verifica dell'Integrità (Forensics):** Il comando **`Get-FileHash`** è fondamentale. Permette di calcolare istantaneamente l'impronta digitale (hash) di un file. Un analista lo usa per **confrontare file sospetti** con database globali di malware, automatizzando una verifica che richiederebbe molto più tempo con strumenti grafici.

2. **Monitoraggio Avanzato della Rete (Threat Hunting):** Mentre **`netstat`** è utile, il cmdlet **`Get-NetTCPConnection`** permette di **filtrare i risultati** via script. Ad esempio, è possibile scrivere una riga di comando che mostra solo le connessioni dirette verso indirizzi IP esteri, aiutando a individuare tentativi di esfiltrazione di dati in tempo reale.

3. **Analisi dei Log di Sistema (Incident Response):** Invece di scorrere migliaia di eventi manualmente, il comando **`Get-WinEvent`** permette di **estrarre** in pochi secondi **tutti i tentativi di accesso falliti** (Event ID 4625). Automazione molto utile per **identificare attacchi di tipo Brute Force** in corso.

In conclusione dell’esercizio 1

L'esperienza pratica con i PID e il networking, unita a questa ricerca, conferma che la padronanza di PowerShell è un requisito essenziale per la sicurezza moderna.

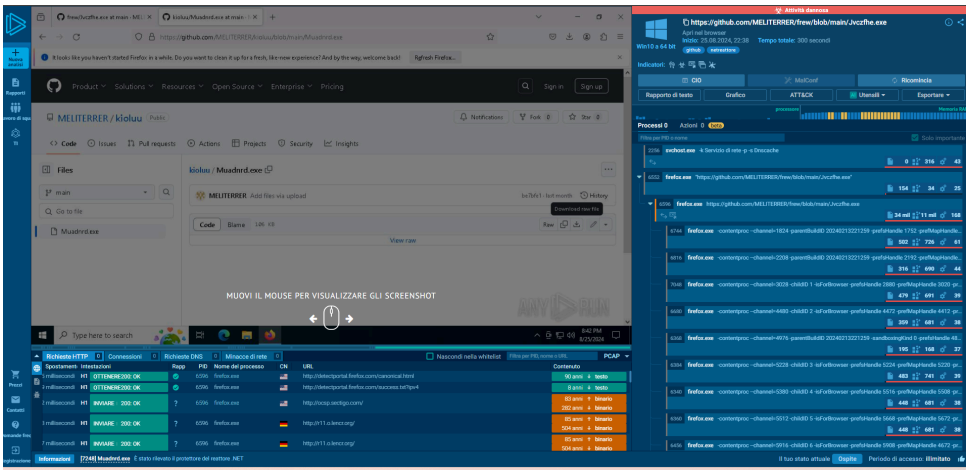
La capacità di passare rapidamente dall'identificazione di una connessione (come con il **PID 1652**) all'automazione della manutenzione (come nel esempio del cestino) **permette di ridurre drasticamente i tempi di risposta agli incidenti**, **garantendo una difesa proattiva** e non solo reattiva.

2. Analisi Ioc su ANY.RUN

In questo report analizziamo una serie di minacce informatiche identificate tramite l'analisi di **Indicatori di Compromissione (IoC)**. Lo scopo è identificare le tracce lasciate dagli attaccanti per prevenire futuri incidenti.

Strumento utilizzato:

- **ANY.RUN (Sandbox Interattiva):** Una piattaforma di analisi dinamica che permette di eseguire campioni di malware in un ambiente Windows isolato e sicuro. Questo strumento consente di **osservare in tempo reale** le modifiche al sistema, le connessioni di rete e la gerarchia dei processi generati dal file malevolo.



2.1 Analisi delle Minacce Rilevate

Dall'analisi dei dati sono emerse tre minacce distinte che operano con modalità differenti:

1. Trojan Downloader (Analisi Comportamentale)

- **Identificazione:** Il file sospetto **`Jvczfhe.exe`** è stato scaricato tramite browser da un repository pubblico (GitHub).
- **Tecnica rilevata:** Il malware utilizza una tecnica di evasione chiamata **"Living off the Land"**. Invece di eseguire codice palesemente maligno, "possiede" il processo di sistema legittimo **`installutil.exe`** per operare nell'ombra e bypassare i controlli dell'antivirus.

2. Campagne di Phishing e URL Malevoli

- **Identificazione:** È stata rilevata una lista di URL classificati come "Attività dannosa".
- **Esempio:** L'URL punta a una pagina di phishing progettata per il furto di identità o di credenziali bancarie.

3. SQL Injection (Violazione Database)

- **Identificazione:** Analisi di hash di password estratti da un database vulnerabile.
- **Dettaglio:** L'uso di algoritmi deboli ha permesso di ottenere l'hash MD5 **`8d3533d75ae2c3966d7e0d4fcc69216b`** (appartenente all'utente "pablo"), che risulta facilmente decifrabile.

MD5: 00B5E91B42712471CDFBDB37B715670C

2.2 Analisi Tecnica degli IoC (Hash)

Per ogni minaccia sono state estratte le "impronte digitali" (Hash), essenziali per la difesa proattiva.

- **MD5:** Utile per identificazioni rapide in database storici.
- **SHA256:** Lo standard attuale per la sicurezza. Un hash come quello trovato nella lista IoC permette di bloccare il file ovunque, indipendentemente dal nome che gli

viene dato.

2.3 Danni e Rimedi

Danni Potenziali:

1. **Esfiltrazione di dati:** Furto di account e informazioni sensibili tramite Phishing e SQL Injection.
2. **Infezione persistente:** Il Trojan può installare altri malware (Ransomware o Spyware) che rimangono attivi anche dopo il riavvio del PC.

Rimedi Consigliati:

- **Blocco degli IoC:** Inserire gli hash SHA256 e i domini rilevati nelle "Blacklist" dei sistemi di protezione aziendali.
- **Sanitizzazione del Codice:** Proteggere i siti web dalle SQL Injection tramite il filtraggio dei parametri di input.
- **Miglioramento del Hashing:** Sostituire l'uso di MD5 con algoritmi più complessi (SHA256 o bcrypt) per proteggere le password degli utenti.

2.4 Conclusione

L'analisi dimostra che la sicurezza informatica non si basa solamente sul bloccare un file, ma sul capire la catena dell'attacco. Grazie alla sandbox ANY.RUN, è stato possibile trasformare dei semplici file sospetti in **Intelligence azionabile**, identificando hash e domini che possono ora essere usati per proteggere l'intera infrastruttura.

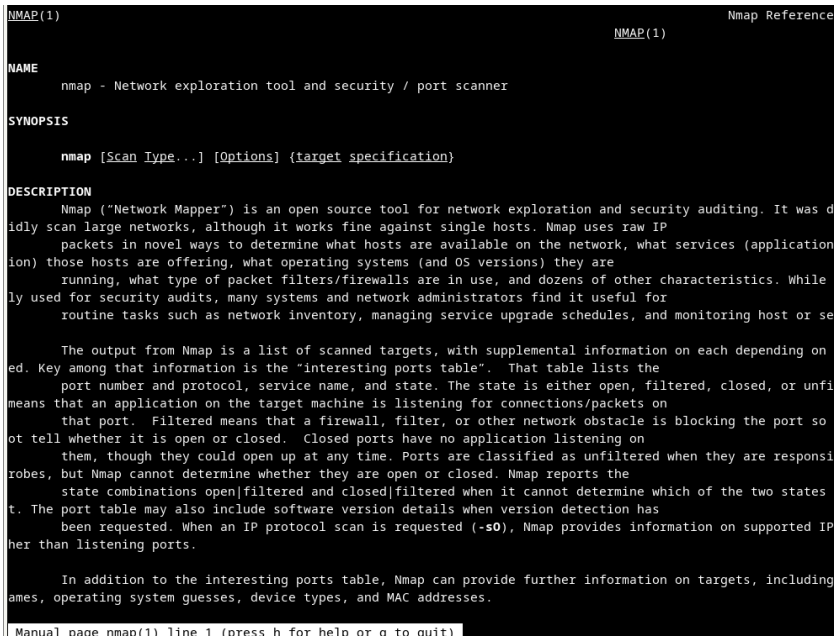
3. Bonus 1: Esplorazione di Nmap

Questa attività si concentra sulla ricognizione di rete tramite **Nmap**, uno strumento fondamentale per il network discovery e l'audit della sicurezza. L'obiettivo è identificare host attivi, porte aperte e servizi in esecuzione per valutare la superficie di attacco.

La macchina virtuale usata in questione è Kali Linux.

3.1 Esplorazione

Per scoprire direttamente dal sistema che cos'è **nmap** utilizziamo il comando: **man nmap**



Domanda: Cos'è Nmap?

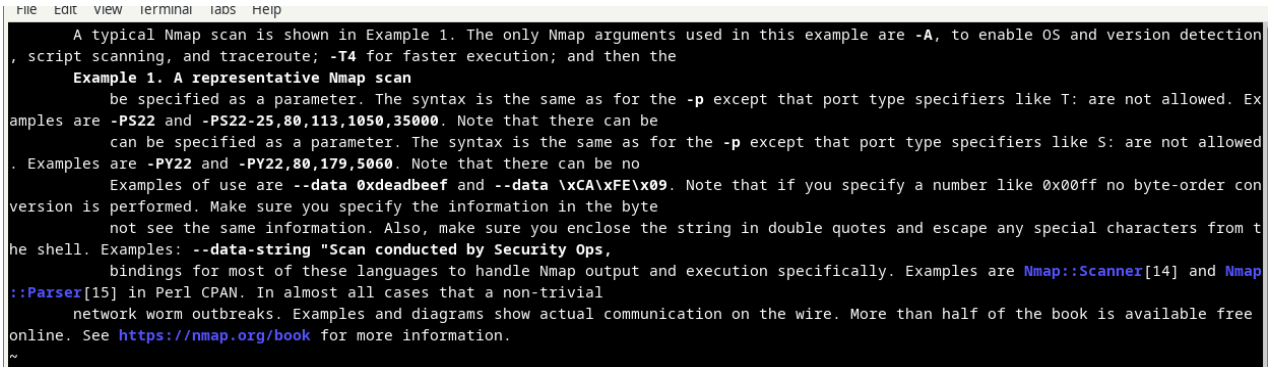
Nmap (Network Mapper) è una potente utility di rete open source.

Domanda: Per cosa viene usato nmap?

Permette di identificare quali host sono attivi su una rete, quali servizi (nome e versione dell'applicazione) offrono, quali sistemi operativi utilizzano e che tipo di firewall o filtri di pacchetti sono in uso.

Per consultare il manuale ufficiale direttamente dal terminale utilizziamo il comando: **/Example.**

Questa operazione istruisce il terminale a evidenziare tutte le occorrenze della parola 'example' nel testo. Saltando la selezione manuale, e ci vengono mostrati i comandi pratici e le combinazioni di opzioni più comuni.



Domanda: Qual è il comando nmap usato?

`nmap -A scanme.nmap.org`

Domanda: Cosa fa l'opzione -A?

L'opzione **-A** abilità diverse funzioni avanzate contemporaneamente per ottenere una scansione completa come:

- Rilevamento del sistema operativo
- Scansione delle versioni
- Scansione tramite script
- Traceroute

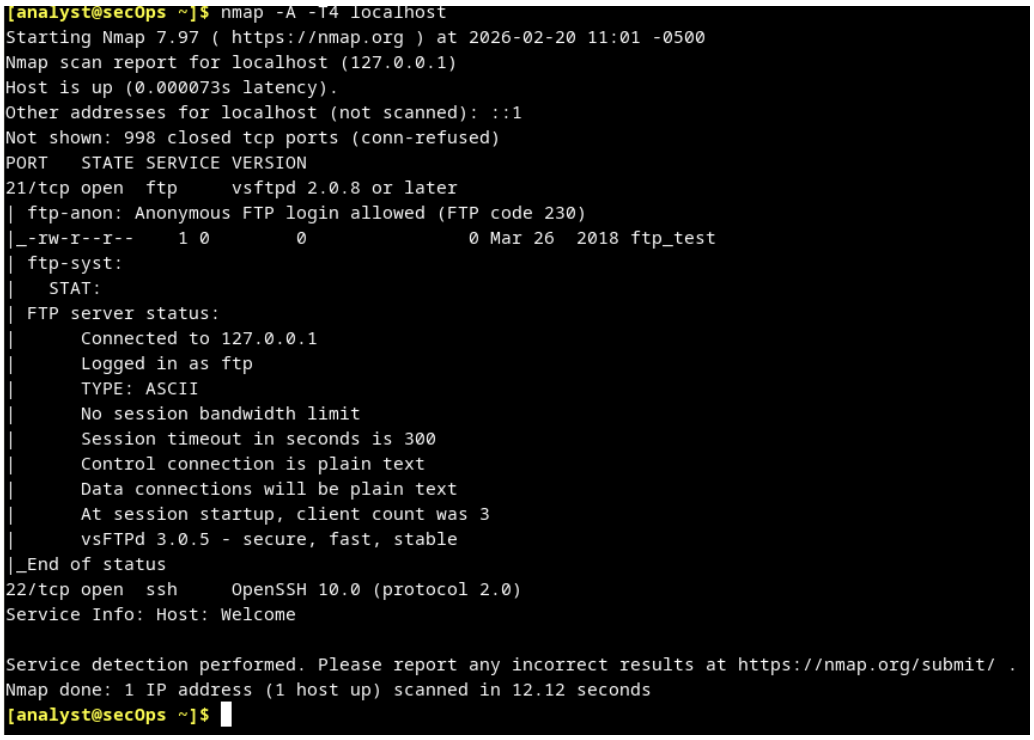
Domanda: Cosa fa l'opzione -T4?

L'opzione **-T4** serve a impostare il "timing template", ovvero la **velocità della scansione**.

- Nmap ha livelli da 0 a 5. più è veloce più aumenta la probabilità di essere individuati e quindi non passare inosservati.

3.2 Scansione delle porte aperte

Applichiamo i comandi studiati per scansionare la macchina locale (localhost).L'obiettivo è verificare quali servizi sono attivi e accessibili direttamente sul nostro sistema con il comando: **nmap -A -T4 localhost**



Domanda: Quali porte e servizi sono aperti?

Dalla scansione effettuata su **`localhost`**, forniamo una analisi del risultato:

- **porta 21/tcp:** ci fornisce lo stato: **open** il servizio: **ftp** e la versione: **vsftpd 2.0.8 or later**
- **porta 22/tcp:** ci fornisce lo stato: **open** il servizio **ssh** e la versione **openSSH 10.0**

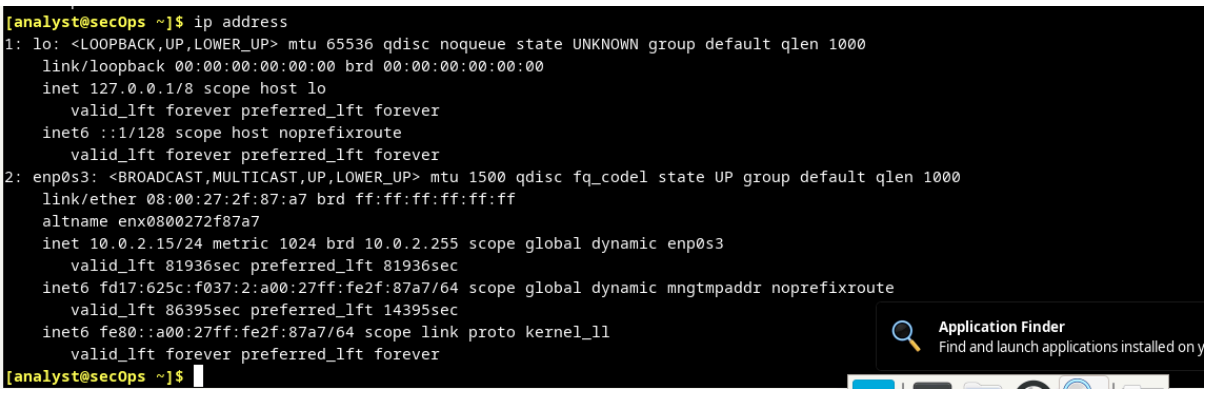
la scansione del localhost ha identificato due servizi attivi: **FTP** e **SSH**.

Grazie all'opzione **-A**, è stato possibile determinare le versioni esatte dei software (**vsftpd 2.0.8** e **OpenSSH 10.0**), fornendo informazioni preziose per la valutazione della sicurezza della macchina locale.

3.3 Scansione della rete

Per procedere con la scansione della rete locale, il primo passo fondamentale è identificare le coordinate della propria macchina all'interno dell'infrastruttura virtuale.

possiamo utilizzare il comando **ip address** da terminale per visualizzare le interfacce di rete, fornendo dettagli molto precisi su ogni scheda di rete.



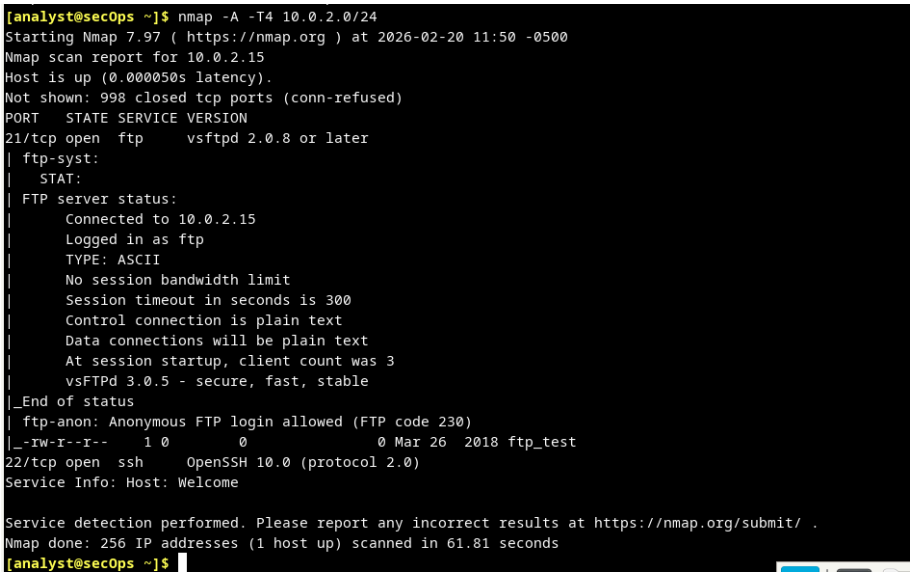
Domanda:A quale rete appartiene la tua VM?

In base alla analisi della rete in cui è presente la macchina sottostante ci fornisce i seguenti dati:

- **Interfaccia di rete:** `enp0s3`.
- **Indirizzo IP della VM:** `10.0.2.15`.
- **Maschera di sottorete:** `/24` (che equivale a `255.255.255.0`).
- **Indirizzo di Rete:** **`10.0.2.0/24`**.

quindi la Virtual Machine appartiene alla rete **`10.0.2.0/24`**

Per localizzare altri host su questa LAN, inseriamo **nmap -A -T4 10.0.2.0/24 ,** questo comando è una scansione completa e ottimizzata rivolta a un'intera sottorete locale, ovvero tutti i 256 indirizzi da 10.0.2.0 a 10.0.2.255



Domanda:Quanti host sono attivi?

Innanzitutto i risultati della scansione di rete **10.0.2.0/24** è presente **1 host up** ovvero la stessa macchina kali Sulla quale sono stati identificati due servizi principali: un server FTP

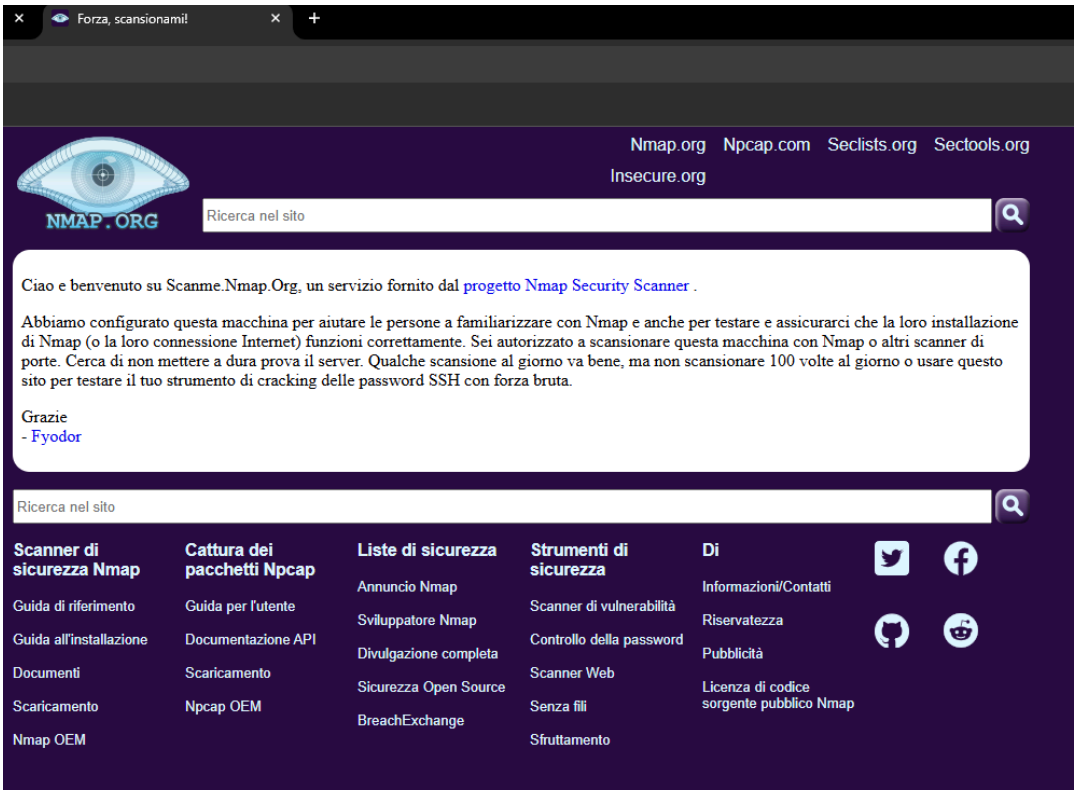
(**vsftpd**) e un server SSH (**OpenSSH**). L'analisi conferma la presenza di una configurazione insicura sul servizio FTP, che consente l'accesso senza credenziali (Anonymous login).

essendoci un solo host la scansione è identica alla scansione localhost ma la differenza tra le scansioni è la seguente:

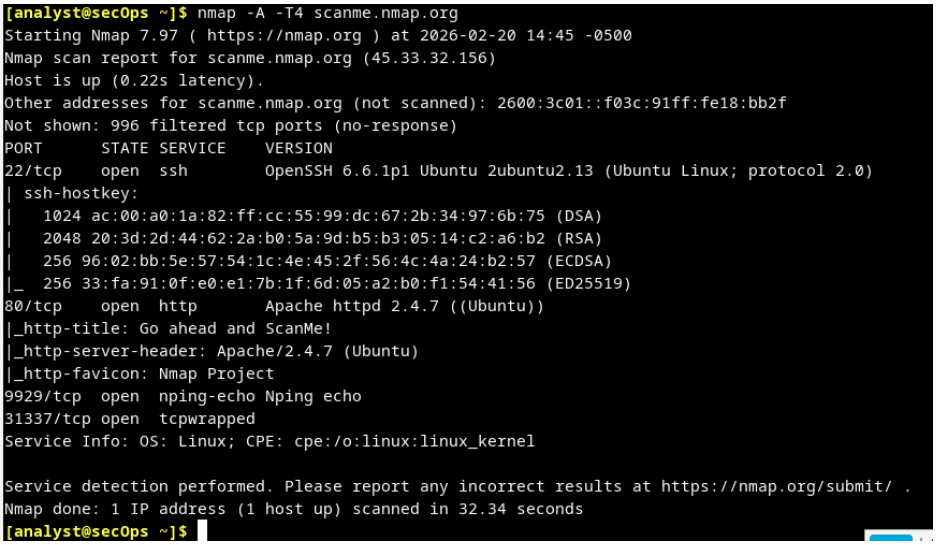
1. La scansione **localhost** è più veloce perché il traffico non passa "fisicamente" attraverso i protocolli di rete esterni della VM
2. La scansione **10.0.2.0/24** serve invece a vedere se ci sono altre persone o dispositivi nella stanza virtuale

3.4 Scansiona un server remoto

Per andare a scansionare un vero e proprio server abbiamo bisogno del permesso del proprietario, quindi ci appoggeremo ad un server pensato apposta per questa evenienza ed imparare ad utilizzare gli strumenti, quindi il sito si presta e quasi impone di essere scannerizzato.



Quindi passiamo all'operazione di scansione vera e propria con il seguente comando da terminale: **nmap A T4 scanme.nmap.org**



Quali porte e servizi sono aperti?

Dall'output risultano **4 porte aperte**:

1. **22/tcp**: Servizio **ssh** (Software: **OpenSSH 6.6.1p1 Ubuntu**).
2. **80/tcp**: Servizio **http** (Software: **Apache httpd 2.4.7**).
3. **9929/tcp**: Servizio **nping-echo** (Software: **Nping echo**).
4. **31337/tcp**: Servizio **tcpwrapped**.

Quali porte e servizi sono filtrati?

- Nmap indica che ci sono **996 porte "filtered"** (nello specifico, porte TCP che non hanno fornito risposta). Questo accade solitamente a causa della presenza di un firewall che scarta i pacchetti senza rispondere.

Qual è il sistema operativo?

- Il sistema operativo rilevato è **Linux**. Nmap specifica che si tratta di una distribuzione **Ubuntu** (dedotto dalle versioni di Apache e OpenSSH) con **kernel Linux**.

Come può Nmap aiutare con la sicurezza della rete?

Nmap è uno strumento fondamentale per la difesa della rete perché permette di:

- **Mappare la superficie di attacco**: Identificare tutti i dispositivi connessi e i servizi esposti.
- **Audit di sicurezza**: Verificare se le versioni dei software installati sono obsolete o vulnerabili a exploit noti.
- **Verifica delle regole del Firewall**: Controllare se le porte che dovrebbero essere chiuse o filtrate sono effettivamente protette.

- **Individuazione di host "ombra"**: Trovare dispositivi non autorizzati collegati alla rete aziendale.

Come può Nmap essere usato da un attore malevolo?

Un hacker può utilizzare Nmap come strumento di ricognizione per:

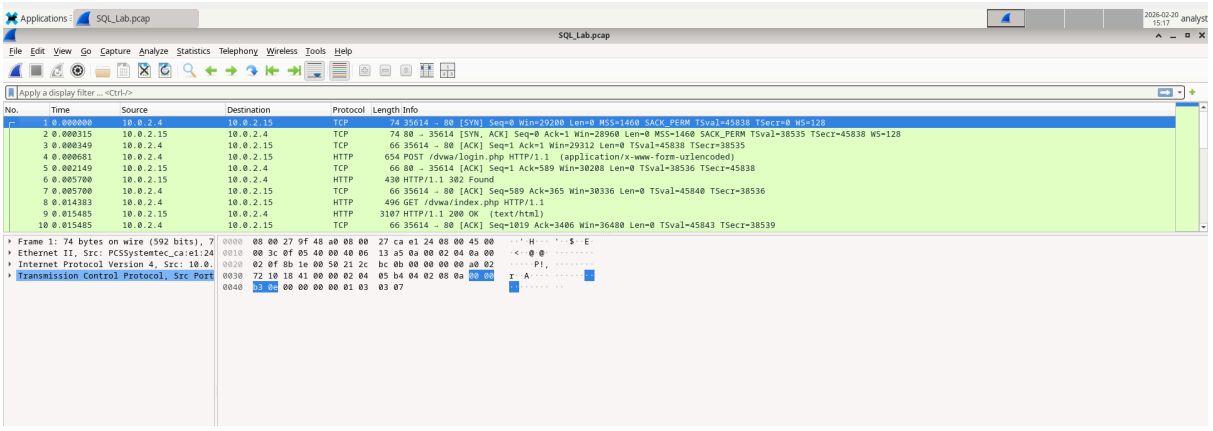
- **Scansione dei bersagli**: Trovare punti di ingresso (porte aperte) in un sistema senza dover interagire direttamente con l'utente.
- **Fingerprinting per attacchi mirati**: Conoscere l'esatta versione di un software (es. Apache 2.4.7) permette di cercare exploit specifici già pronti per quel servizio.
- **Ricerca di vulnerabilità note**: Utilizzare script automatici (NSE) per trovare falle come l'accesso FTP anonimo o database non protetti.
- **Evasione e mascheramento**: Configurare la velocità e il tipo di pacchetti per cercare di non far scattare i sistemi di allarme (IDS) della vittima.

In definitiva, l'attività prova che la sicurezza non dipende solo dalla chiusura delle porte, ma dalla corretta configurazione dei servizi attivi e dal monitoraggio costante di ciò che è visibile dall'esterno.

4. Bonus 2: Attacco a un database MySQL

In Questo laboratorio analizzeremo un attacco SQL injection precedentemente catturato in un file PCAP, useremo il programma wireshark per l'analisi.

Gli attacchi di SQL injection consentono agli hacker malintenzionati di digitare istruzioni SQL in un sito web e ricevere una risposta dal database.



Domanda: Quali sono i due indirizzi IP coinvolti in questo attacco di SQL injection in base alle informazioni visualizzate?

Dall'analisi del traffico di rete catturato, si identificano chiaramente i due attori della comunicazione:

1. **Sorgente (Attaccante): `10.0.2.4`**

- Questo indirizzo IP avvia la connessione TCP (pacchetto 1, flag `[SYN]`) e invia le richieste HTTP POST e GET.
- È l'indirizzo da cui originano le query dirette verso le pagine sensibili come **`/dvwa/login.php` e `/dvwa/index.php`.**

1. **Destinazione (Vittima/Server): `10.0.2.15`**

- Questo è l'indirizzo IP che ospita l'applicazione web (DVWA - Damn Vulnerable Web Application).
- Risponde alle richieste dell'attaccante confermando la ricezione dei dati e restituendo i contenuti delle pagine (pacchetti `200 OK` o `302 Found`).

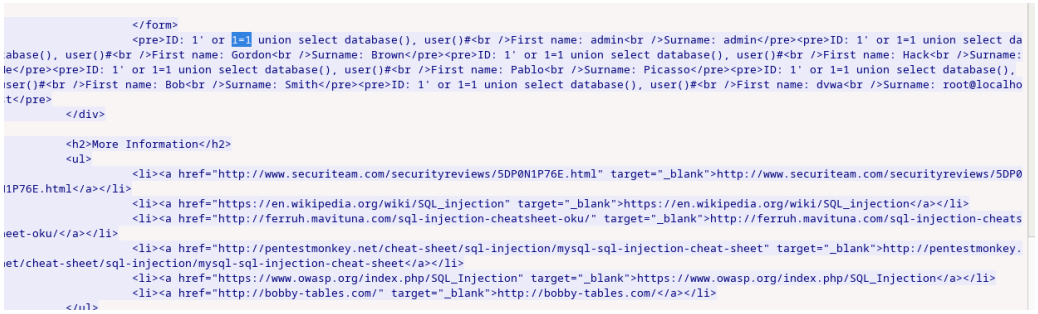
Seguendo il flusso HTTP dell'attaccante vediamo i passi che l'aggressore svolge.

1. **Visualizzare l'attacco di SQL Injection**

   L'attaccante ha inviato una query (**1=1**) in una casella di ricerca UserID sulla vittima, Invece di rispondere con un messaggio di fallimento del login, l'applicazione ha risposto con un record da un database, così facendo ha scoperto la vulnerabilità del SQL injection.

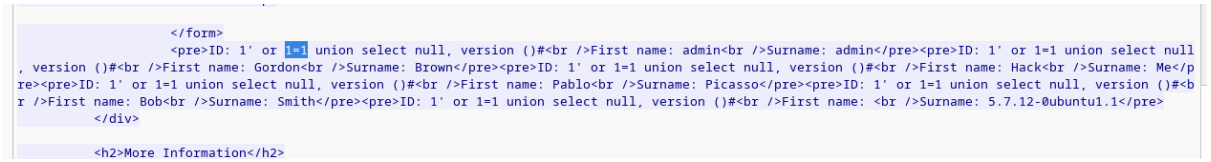
2. **L'attacco continua**

   Da qui l'aggressore ha inserito una query (**1' or 11 union select database(), user()#**) e estrae altre informazioni come il nome del database **dvwa,** l'utente del database **root@localhost** e altri account utente.



3. **L'attacco fornisce informazioni di sistema**

   L'aggressore ha inserito una query 1' or 11 union select null, version ()#) per individuare l'identificatore di versione.



Domanda: Qual è la versione?

La versione che scopre l'aggressore del Database è **5.7.12-0ubuntu1.1**, informazioni estremamente pericolose per il riconoscimento dei punti deboli e scoprire sia il sistema che la versione delle tabelle contenente i dati sensibili.

4. **L'attacco di SQL Injection e le informazioni sulle tabelle**

   L'attaccante inserisce la query (1'or 11 union select null, table_name from information_schema.tables#) anche se gli fornisce un output enorme di molte cartelle, poiché l'attaccante ha inserito "null" senza ulteriori specifiche.

Cosa farebbe per l'aggressore il comando modificato di 1′ OR 11 UNION SELECT null, column_name FROM INFORMATION_SCHEMA.columns WHERE table_name='users'?

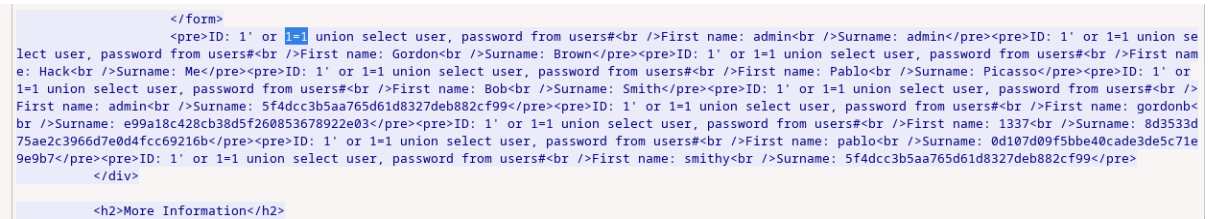
Il comando (**1' OR 1=1 UNION SELECT null, column_name FROM INFORMATION_SCHEMA.columns WHERE table_name='users'#**) permetterebbe all'aggressore di **estrarre i nomi di tutte le colonne** della tabella chiamata **users.**

Il database risponderebbe con un output molto più breve, filtrato per l'occorrenza della parola "**users"**.

5. **L'attacco di SQL Injection si conclude**

   L'aggressore ha inserito una query 1'or 11 union select user, password from users#) in una casella di ricerca UserID sulla vittima. così facendo questo comando ha

forzato il database a stampare il contenuto delle colonne **`user`** e **`password`** della tabella **`users`**, anche se l'output non restituisce le password in chiaro comunque rende gli hash delle password che sono facilmente decifrabili.

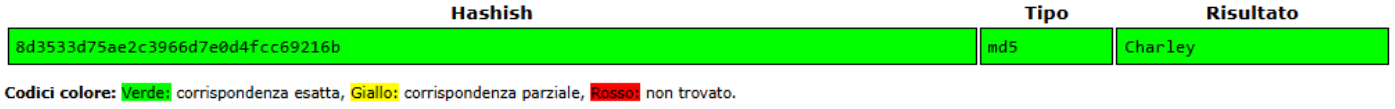


Quale utente ha l'hash della password di 8d3533d75ae2c3966d7e0d4fcc69216b?

Questo hash appartiene all'utente con il firstname: **1337**.

Qual'è la password in chiaro?

Convertendo l'hash della password scopriremo che la password è **charley**, probabilmente il nome dell'utente.



Qual è il rischio che le piattaforme utilizzino il linguaggio SQL?

Il rischio principale non è il linguaggio SQL in sé, che è uno standard fondamentale, ma la **mancata sanificazione degli input**.

se un'applicazione "si fida" ciecamente di ciò che l'utente scrive nei form (come il campo ID o login), un attaccante può inserire comandi SQL arbitrari.

Quindi non filtrare l'input potrebbe causare eventuali problematiche come:

- Accesso non autorizzato
- Perdita di riservatezza
- Intercettazione dati

Una volta trovata la vulnerabilità, l'aggressore può decidere il livello di danno e può agire come:

- aggressore opportunista
- attore malevolo professionista
- O causare danno distruttivo

Per prevenire gli attacchi di SQL injection, è fondamentale adottare una strategia di difesa a più livelli.

Quali sono 2 metodi o passaggi che possono essere adottati per prevenire gli attacchi di SQL injection?

Ecco due dei metodi principali che possono essere implementati:

1. **Utilizzo di Query Parametrizzate (Prepared Statements):**

   Invece di inserire direttamente l'input dell'utente nella stringa della query SQL, si utilizzano dei segnaposto (parametri). Il codice SQL viene inviato al database separatamente dai dati dell'utente; in questo modo, il database tratta l'input esclusivamente come testo e non come parte del comando eseguibile, rendendo impossibile l'iniezione di codice malevolo

2. **Implementazione di un Web Application Firewall (WAF):**

   Un WAF fa da scudo tra l'applicazione web e Internet. Monitora e filtra il traffico HTTP in entrata, utilizzando un elenco di "firme" o regole costantemente aggiornate per identificare e bloccare i pattern tipici degli attacchi SQL injection prima che raggiungano il database

   altre prevenzioni includono:

   - **Filtrare e convalidare l'input dell'utente:** Accettare solo dati che corrispondono a un formato previsto
   - **Utilizzare i parametri con le Stored Procedure:** Simile alle query parametrizzate, le stored procedure memorizzate nel database dovrebbero essere scritte in modo da non concatenare dinamicamente l'input dell'utente all'interno del codice SQL
   - **Principio del minimo privilegio**
   - **Disabilitare funzionalità non necessarie**