Hydra

1. Introduction

Hydra (also called THC-Hydra) is a powerful online password-cracking tool used to test login forms and network services. Unlike Hashcat or John (which crack offline hashes), Hydra tests live authentication services (like SSH, FTP, HTTP)to identify weak credentials and improve security.

Note: For tetsing, I am using demo site "http://testphp.vulnweb.com/" and metasploitable 2 VM (192.168.1.76)

2. Basic hydra syntax

- **Command:** *hydra -l <user> -P <passwordlist> <service>://<target>*
- **Explanation: -**l sets a username, -P sets a wordlist, and the service and target define where Hydra should attempt logins. This is the fundamental structure used for most tests.

3. Different services testing

- **SSH Login**

→ **SSH:** a cryptographic network protocol that provides a secure, encrypted channel for remote operations over an unsecured network. Its key functions include enabling secure remote login and execution of commands on servers (e.g., Linux/Unix administration), protecting all data transfer (including passwords) from eavesdropping via strong encryption, and facilitating secure file transfers through protocols like SFTP and SCP. It is the modern standard for remote system access, relying on either password or the more secure key-based authentication.

→ **Command:** *hydra -L user.txt -P passowrds.txt ssh://<target_ip>*

→ **Explanation:** Tests SSH logins using a wordlist.

- **FTP login**

→ **FTP:** a standard network protocol dedicated solely to transferring and managing files between a client and a server. It is a highly effective, yet insecure, protocol because it sends all communication, including user credentials (username and password) and the files themselves, in unencrypted plain text over the network. This fundamental lack of security makes it vulnerable to interception, which is why it has largely been replaced by secure alternatives like SFTP (SSH File Transfer Protocol), which uses SSH to encrypt the entire communication channel.

→ **Command:** *hydra -l msfadmin -P /home/paxton/usernamesPassword.txt ftp://192.168.1.76*

→ **Explanation:** Attempts FTP authentication using a given user and wordlist. This helps verify that old services like FTP are properly secured or disabled.

```
┌──(paxton㉿kali)-[~]
└─$ hydra -l msfadmin -P /home/paxton/usernamesPassword.txt ftp://192.168.1.76
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secr
et 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 2025-12-01 12:48:10
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a
 previous session found, to prevent overwriting, ./hydra.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 19 login tries (l:1/p:19), ~2 tries per ta
sk
[DATA] attacking ftp://192.168.1.76:21/
[21][ftp] host: 192.168.1.76   login: msfadmin   password: msfadmin
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-12-01 12:48:24
```

- **HTTP post login**

→ **Command:** *hydra -l admin -P pass.txt 192.168.1.10 http-post-form "/login.php:username=^USER^&password=^PASS^:Invalid"*

→ **Explanation:** Sends POST requests to login page with dynamic user/pass placeholders. We use this when testing web login strength on website.

- **Https(ssl) post login**

→ **Command:** *hydra -l admin -P pass.txt 192.168.1.10 https-post-form "/login:usr=^USER^&pwd=^PASS^:error"*

→ **Explanation:** Same as HTTP but for HTTPS. We use it to test secure pages and confirm rate-limiting is active.

- **RDP (remote desktop) login**

→ **RDP:** Remote Desktop Login allows a user to access and control a remote computer's graphical desktop environment as if they were seated locally, with the most common protocol being RDP (Remote Desktop Protocol), which typically uses TCP Port 3389. The process involves a client (your machine) establishing an encrypted connection to the server (the target machine) after authenticating with the user's credentials, after which the client transmits screen updates and keyboard/mouse inputs, giving the user full administrative control over the remote system.

→ **Command:** *hydra -t 4 -V -l admin -P passwords.txt rdp://192.168.1.20*

→ **Explanation:** Tests RDP logins using 4 threads (safe for lab). It helps you ensure remote access is secure and not using default passwords.

- **MYSQL database login**

→ **Command:** *hydra -l root -P passwords.txt mysql://<target_ip>*

→ **Explanation:** Attempts MySQL authentication. Useful for verifying database security and ensuring no weak root passwords exist.

4. Hydra speed & thread control

- **Control thread count (prevent service crash)**

→ **Command:** *hydra -l admin -P pass.txt -t 4 ssh://192.168.1.10*

→ **Explanation:** -t limits threads to avoid overloading your service. This is important so your lab server doesn't freeze during testing.

- **Slow mode (Bypass weak rate limits)**

→ **Command:** *hydra -l admin -P pass.txt -t 1 ssh://192.168.1.10*

→ **Explanation:** Uses one thread to emulate slow, human-like logins. Useful when testing systems that lock out after too many fast attempts.

5. Use a Proxy or Tor

- **Routing traffic through proxy**

→ **Command:**

→ **Explanation:** Routes Hydra traffic through a proxy for analysis. Helps you understand how firewalls detect or fail to detect proxied logins.

- **Using Tor**

→ **Command:**

→ **Explanation:** Forces Hydra traffic through Tor SOCKS5. This teaches you how anonymity networks interplay with login attempts, strictly in a controlled environment.