THC-Hydra — Professional Cheat Sheet

One-page professional reference for `hydra` (THC-Hydra). Fast, parallelized network login cracker supporting many protocols. Commands, flags, operational patterns, examples and OPSEC notes for red-team and pentest use.

---

1) At-a-glance

- **Tool:** `hydra` (THC-Hydra) — parallelized network login cracker for online brute-force and dictionary attacks.
- **Use cases:** Credential testing for services (SSH/FTP/HTTP/POP3/SMB/etc.), assessing password policy strength, credential stuffing (with permission).
- **Warning:** Online attacks are noisy, may trigger lockouts/IDS, and are illegal without explicit authorization. Use only on assets you own or have written permission to test.

---

2) Install / quick check

```
# Debian/Ubuntu (Kali includes it)
sudo apt update && sudo apt install hydra

# From source (upstream)
git clone https://github.com/vanhauser-thc/thc-hydra.git
cd thc-hydra
./configure && make && sudo make install

# Check available modules and version
hydra -h       # summary
hydra -U <service>  # show module-specific options
hydra -L userlist -P passlist target ssh://host
```

---

3) Core options (accurate & concise)

- `-l LOGIN` : single username
- `-L FILE` : file with usernames (one per line)
- `-p PASS` : single password
- `-P FILE` : password list file (wordlist)
- `-x MIN:MAX:CHARSET` : generate passwords (bruteforce). CHARSET: `1` digits, `a` lowercase, `A` uppercase; you may include other characters literally (e.g. `1:3:a1%`).
- `-y` : disable symbols when using `-x`

- `-e nsr` : extra checks — `n` null password, `s` try username as password, `r` try reverse(username) as password
- `-C FILE` : colon-separated `user:pass` combos (combo file)
- `-u` : loop passwords across all users (password-first strategy)
- `-f` : stop after first valid pair per host (use with single host or -M)
- `-F` : stop after first valid pair for any host (with -M)
- `-M FILE` : attack multiple targets (one host per line)
- `-o FILE` : write found credentials to FILE
- `-b FORMAT` : output format for -o (text, json, jsonv1)
- `-t TASKS` : number of parallel tasks (default 16)
- `-w TIME` : max wait time for responses (seconds, default 32)
- `-W TIME` : wait between each connection a task performs
- `-c TIME` : wait time per login attempt across all threads
- `-s PORT` : custom port
- `-S` : use SSL/TLS
- `-4` / `-6` : prefer IPv4 / IPv6
- `-v` / `-V` : verbose modes
- `-d` : debug mode
- `-R` : restore previous session from hydra.restore

---

4) Common protocols & modules

Hydra supports many protocols: `ssh`, `ftp`, `smtp`, `http-get-form`, `http-post-form`, `https-post-form`, `rdp`, `svn`, `vnc`, `mysql`, `mssql`, `oracle`, `redis`, `smtp-enum`, `sshkey`, and dozens more. Use `hydra -U <module>` to view module options. (Exact list depends on build and compile-time libraries.)

---

5) Practical examples

5.1 SSH (single user + wordlist)

```
hydra -l root -P /usr/share/wordlists/rockyou.txt -t 4 ssh://10.10.10.5 -f -o hydra_found.txt
```

- `-t 4` keeps concurrency conservative. `-f` stops after first found for that host.

5.2 SSH across many hosts (parallel targets)

```
hydra -L users.txt -P passwords.txt -M hosts.txt -t 64 ssh -o results.txt
```

- `-M` reads hosts; Hydra parallelizes across targets. Monitor load and network limits.

5.3 HTTP POST form (typical web login)

```
hydra -l alice -P /usr/share/wordlists/passwords.txt 10.10.10.20 http-post-form "/login.php:username=^USER^&password=^PASS^:F=incorrect" -t 10 -V -o hydra_http.txt
```

- Replace `F=` token with part of the response that indicates a failed login (e.g. "Invalid user or password"). Use `-V` for verbose attempts while testing syntax.

5.4 HTTPS form

```
hydra -L users.txt -P pass.txt 192.168.1.30 https-post-form "/auth:email=^USER^&pwd=^PASS^:F=Login failed" -s 443 -S -t 8
```

5.5 Using combo files (user:pass)

```
hydra -C combos.txt -M hosts.txt ssh -t 8 -o combos_found.txt
```

5.6 Brute-force generation example (small demo)

```
# generate passwords length 1..3 with lowercase and digits
hydra -l admin -x 1:3:a1 -t 8 ssh://10.0.0.5
```

- Use carefully — exponential blowup. Prefer dictionary lists for real testing.

---

6) Tuning for reliability & stealth

- **Respect account lockouts:** set low parallelism (`-t 1..4`) and use `-W` / `-c` to insert delays if the target enforces rate limits.
- **Avoid IDS signature spikes:** randomize timing and rotate source IPs (careful with legal implications).
- **Use small bursts:** `-t` smaller and `-W` non-zero reduce noise.
- **Use `-f` with caution:** stops on first valid per host — useful in large scans.

---

7) Operational patterns & helpers

- **Proxy/Tor chaining:** run Hydra through `proxychains` or a SOCKS proxy. Note Tor exit nodes are slow and often block many services.
- **Combining with `xhydra`:** GUI front-end for visual config (if installed).

- **Session resume:** hydra creates `hydra.restore` — use `-R` to continue.
- **Module options:** `hydra -U <service>` to list extra options a module accepts (e.g., HTTP headers, form fields).

---

8) Detection & logging considerations

- Brute force attempts produce obvious logs (auth failures) — coordinate with your client and monitoring teams.
- Use IDS/IPS logs to measure impacts; capture pcap for evidence.
- Keep detailed notes of tests (time, targets, accounts tried, wordlists used).

---

9) Alternatives & complementary tools

- **ncrack** — focused on network authentication cracking with different tuning.
- **Medusa** — similar to Hydra with different modules and threading model.
- **Burp Suite Intruder** — better for complex web login flows and CSRF tokens.
- **Custom scripts with** `requests` **/** `paramiko` when target requires more control.

---

10) Quick checklist before running

1. Written authorization / signed scope.
2. Verify target IPs, ports, and service behavior manually.
3. Start with non-destructive tests, low `-t`, and logging enabled.
4. Use appropriate wordlists and `-e` options to try null/username/reverse tests.
5. Clean up, provide artifacts (logs, results), and report responsibly.

This cheat sheet is for legal pentesting and educational use only. Do not use `hydra` *on systems you do not own or have explicit permission to test.*