HackMate CLI Assistant: Comprehensive User Manual

Author: Manus AI **Date:** November 10, 2025 **Version:** 1.0 (Final Debugged Release)

1. Introduction to HackMate

HackMate is a modular, extensible command-line assistant designed to streamline and automate common tasks for penetration testers, red-teamers, and bug-bounty hunters operating on Kali Linux. It serves as a consistent interface over a collection of industry-standard tools, enforcing a structured workflow and prioritizing safety and auditable results.

The primary goals of HackMate are to:

- **Automate repetitive tasks** (e.g., reconnaissance, scanning).
- **Enforce safe and ethical testing practices** through explicit scope confirmation.
- **Organize results** into per-target workspaces for easy reporting and triage.
- **Provide a foundation** for advanced automation via YAML-based flows and AI assistance.

2. Installation and Setup

This guide assumes the user is operating on a Kali Linux or Debian-based system.

2.1 Prerequisites

The following software and dependencies are required for full functionality:

- **Operating System:** Kali Linux (Recommended) or Debian/Ubuntu.
- **Python:** Python 3.10 or newer.
- **External Tools:** `nmap`, `masscan`, `ffuf`, `whatweb`, `searchsploit`, `subfinder`, and `httpx`.

2.2 Installation Procedure

The simplest and most reliable method for installation is using the provided `install.sh` script.

1. **Extract the Archive:** First, extract the contents of the final release archive: ```bash tar -xzvf hackmate_final_release.tar.gz cd hackmate ```
2. **Run the Installer:** The installer requires root privileges to install system packages and create the global executable link. ```bash sudo bash install.sh ```

The script will perform the following actions:

- Install Python dependencies (e.g., `click`, `rich`, `tinydb`).
- Check for and install required external tools via `apt` (e.g., `nmap`, `masscan`).
- Create a global symbolic link (`/usr/local/bin/hackmate`) for easy command-line access.

2.3 Configuration

HackMate manages its settings via a YAML configuration file located in the user's home directory: `~/.hackmate/config.yaml`.

To view the current configuration and workspace root: ```bash hackmate config ```

The configuration allows for customization of tool paths, concurrency limits, and safety defaults.

3. Safety and Ethical Use

HackMate is built with a **safety-by-default** philosophy. All users are ethically and legally responsible for their actions.

3.1 Mandatory Safety Controls

Two flags are mandatory for intrusive or scope-sensitive operations:

Flag	Purpose	Commands Requiring Flag
`--confirm-scope`	Explicitly confirms that the user has obtained **written authorization** to test the target. Required for all flows and reconnaissance commands.	`recon subdomains`, `scan masscan`, `scan nmap`, `flow run`
`--execute`	Explicitly confirms that the user is authorizing an **intrusive or destructive** action (e.g., active scanning, fuzzing).	`scan masscan`, `scan nmap`, `web test --dirs`

> **Warning:** Unauthorized testing is illegal and unethical. HackMate's safety controls are helpful but do not absolve the user of legal responsibility. Always ensure a signed scope document is in place before execution.

4. Core Command Reference

HackMate commands are organized into logical groups.

4.1 Reconnaissance (`hackmate recon`)

Focuses on passive and semi-passive information gathering.

Command	Description	Example
`subdomains <target>`	Uses `subfinder` to find subdomains. Results are saved to `<workspace>/subdomains_raw.txt` .	`hackmate recon subdomains example.com --confirm-scope`
`probe <target>`	Uses `httpx` to check the list of discovered subdomains for live HTTP/S services. Results are saved to `<workspace>/live_hosts_raw.txt` .	`hackmate recon probe example.com`

4.2 Scanning (`hackmate scan`)

Focuses on active network and port scanning.

Command	Description	Example
`masscan <target>`	Performs a fast, wide-range port scan.	`hackmate scan masscan 192.168.1.0/24 --execute --confirm-scope`
`nmap <target>`	Performs a targeted Nmap scan. Use `--fast` or `--full` for pre-defined profiles.	`hackmate scan nmap target.com --ports 80,443 --fast --execute --confirm-scope`

4.3 Web Testing (`hackmate web`)

Focuses on web application enumeration and fuzzing.

Command	Description	Example
`test -u <url> --dirs`	Performs directory brute forcing using `ffuf`.	`hackmate web test -u https://target.com --dirs --execute`
`test -u <url> --cms`	Performs CMS and technology fingerprinting using `whatweb`.	`hackmate web test -u https://target.com --cms`

4.4 Exploitation (`hackmate exploit`)

Provides utility functions for exploit discovery and payload generation.

Command	Description	Example
`search <term>`	Searches the local Exploit-DB database using `searchsploit`.	`hackmate exploit search "wordpress 6.2"`
`shell --reverse`	Generates common reverse shell payloads based on user-provided LHOST and LPORT.	`hackmate exploit shell --reverse --lhost 10.0.0.1 --lport 4444`

5. Workflow and Reporting

HackMate's true power lies in its ability to manage and automate the entire testing workflow.

5.1 Workspace Management

All results are stored in a dedicated, consistent workspace directory for each target.

- **Location:** The default workspace root is `~/.hackmate/workspaces/`.
- **Structure:** Each target (e.g., `example.com`) gets its own directory containing raw tool outputs (`subdomains_raw.txt`), structured artifacts (future JSON files), and reports.

5.2 Notes and Findings (`hackmate notes`)

Findings are stored in a structured database (`~/.hackmate/notes.json`) for easy retrieval and reporting.

1. **Adding a Finding:** ```bash hackmate notes add example.com -t "Reflected XSS" -b "/search?q=PAYLOAD" ```
2. **Listing Findings:** ```bash hackmate notes list example.com ```

5.3 Automated Flows (`hackmate flow`)

Flows allow you to define a sequence of HackMate commands in a YAML file for one-click execution.

1. **Flow File Structure (Example: `quick-recon.yaml`):** ```yaml name: quick-recon description: Passive recon, live host probing, and a quick Nmap scan. steps:
   - recon_subdomains: {}
   - recon_probe: {}
   - scan_nmap: ports: "80,443,8080" fast: true ```
2. **Running a Flow:** ```bash hackmate flow run hackmate/flows/quick-recon.yaml target.com --confirm-scope --execute ```

5.4 Reporting (`hackmate report`)

The reporting module aggregates all notes and lists all artifacts in the workspace to generate a final report.

1. **Generating the Report:** ```bash hackmate report generate target.com --pdf ``` This command creates a Markdown file (`target.com_report.md`) and a placeholder PDF file (`target.com_report.pdf`) in the target's workspace.

6. Advanced Features (AI and Extensibility)

6.1 AI Assistance (Placeholder)

HackMate includes a placeholder for AI integration to suggest next steps based on current findings.

- **Command:** `hackmate flow suggest <target>`
- **Enabling AI:** Edit `~/.hackmate/config.yaml` and set `ai: enabled: true` and provide your `api_key` .

6.2 Plugin System (Placeholder)

The architecture supports a drop-in plugin system (`hackmate/hackmate/plugins/`) for adding custom tool wrappers or external API integrations (e.g., Shodan, VirusTotal).

7. Troubleshooting

Issue	Cause	Solution
`command not found: hackmate`	Global link not created or PATH not updated.	Re-run `sudo bash install.sh`.
`Error: Tool 'subfinder' not found`	External tool is missing or not in PATH.	Manually install the missing tool (e.g., `subfinder`) and ensure it is in a directory listed in your system's PATH.
`ImportError: attempted relative import`	Python environment issue.	The `install.sh` script is designed to fix this by setting `PYTHONPATH`. Re-run the installer.
`Nmap error: cannot use -F with -p`	Incompatible Nmap flags.	This was fixed in the final release. Ensure you are using the latest code from the final archive.
`Safety Error: requires --execute`	Intrusive command was run without the required flag.	Add the `--execute` flag to the command.

References

No external references are cited in this manual, as the content is derived from the HackMate project specification and implementation details.