



KALIRANGE // OFFENSIVE TOOLING

SKIRMISHER

COMMAND REFERENCE

Every command, what it does, and how to use it.

TOOL	VERSION	SHELL	DATE
SKIRMISHER / SK	V1.0.0	ZSH • BASH	06 SEP 2026

01 THE ONE COMMAND

Call it, name the target, go – like `nmap <target>`. A bare target runs the recon pipeline (the default action); no `recon` keyword needed. Everything is `scope-gated` and `audited`: no network action runs against a host that isn't on an explicit, non-expired allow-list.

02 COMMANDS – IN DETAIL

skirmisher <target>

```
skirmisher acme.com
```

The default action – no `recon` keyword needed. Runs the full OSINT pipeline against an authorized target: infers its email/username schema, correlates scattered signals into confidence-scored identities (exact → fuzzy → perceptual avatar-hash), enriches them, maps in-scope authentication surfaces, and returns a ranked "most-likely way in". Accepts several targets at once, and `-` to read them from stdin. The scope is auto-discovered; it refuses to run without one.

skirmisher init [target]

```
skirmisher init -g acme.com
```

Scaffolds a `scope.json` engagement file, pre-filling the allow-list with the target and its subdomains and setting a one-year window. Edit `authorized_by` before you run – the scope is your declaration of authorization. `-g` writes it globally to `~/.config/skirmisher/scope.json` so `skirmisher <target>` then works from any directory; `-o FILE` chooses the path; `--force` overwrites an existing one.

skirmisher recon <target>

```
skirmisher recon acme.com -O -P
```

The explicit form of `skirmisher <target>` – runs the five stages (Profile → Correlate → Enrich → Surface → Rank). Add `-O` for live surface discovery (DNS + TLS on in-scope hosts), `-P` for passive crt.sh subdomain discovery, and `-e FILE` for credential-reuse enrichment. `-j` / `--dot` change the output format.

skirmisher scope [file]

```
skirmisher scope eng.json -H vpn.acme.com
```

Validates a scope file – it must be well-formed, name an `authorized_by`, carry a non-empty allow-list, and not be expired – then prints what's authorized (allow / deny). With `-H <host>` it tests a single host against the scope and prints ALLOW or DENY with the reason, so you can confirm a target is in scope before touching it. The file is auto-discovered if omitted.

skirmisher modules

```
skirmisher modules
```

Lists every available module with its short alias and a one-line description, and whether it is read-only `[aux]` or `[invasive]`. Use the alias with `run` / `check` – e.g. `probe`, `expose`, `headers`, `fp`, `tls`, `ssh`.

skirmisher collectors

```
skirmisher collectors
```

Lists the recon collectors and their tier: offline (always run – schema inference, fixtures), `network` (`-O`: live surface discovery), or `passive` (`-P`: crt.sh). Shows which are gated behind an opt-in flag.

02 COMMANDS – IN DETAIL (CONT.)

skirmisher run <module> [target]

```
skirmisher run expose vpn.acme.com
```

Runs a module against a host. The positional target sets `RHOST`; set other options with `-o KEY=VAL` (repeatable, e.g. `-o RPORT=8443 -o SSL=false`). The runner authorizes every target through the scope guard **before** the module executes, so an out-of-scope host is refused before a single packet is sent – and the decision is audited. Each meaningful result is recorded as a **finding** (to `<scope>.findings.jsonl`) for `skirmisher report`; `--no-findings` skips that.

skirmisher check <module> [target]

```
skirmisher check probe 10.0.0.5
```

Runs a module's non-invasive **check** only – the quick "is it there / is it exposed?" probe without the full action (e.g. which ports are open, whether the host serves HTTP). Same scope-gating, options, and findings behaviour as `run`.

skirmisher report

```
skirmisher report -o report.md
```

Builds a Markdown penetration-test report from the findings recorded during the engagement: an executive summary with a severity tally, the scope, every finding by severity with its evidence and timestamp, a methodology section derived from the audit trail, and an appendix. `-o FILE` writes it to a file (otherwise stdout); `-s` points at the scope for engagement metadata; findings are auto-discovered alongside the scope.

skirmisher completions bash|zsh

```
skirmisher completions zsh
```

Prints a shell-completion script for the `sk` and `skirmisher` commands so subcommands and module aliases tab-complete. Source it from your shell rc – e.g. `skirmisher completions zsh > ~/.config/skirmisher/completion.zsh`, then `source` it (after `compinit` on zsh).

skirmisher man

```
skirmisher man
```

Prints the full manual to the terminal – the same reference as the installed `man sk` page (synopsis, commands, options, pipes, exit status, and the authorization statement).

skirmisher version

```
skirmisher version
```

Prints the installed version. `skirmisher --version` works too.

Every command also accepts `skirmisher <command> -h` to print its full flags. The next pages detail the flags for the commands that have them (recon, run/check) plus scope, output, and examples.

03 RECON OPTIONS – `SKIRMISHER <TARGET> [OPTIONS]`

FLAG	WHAT IT DOES
<code>-s, --scope FILE</code>	Engagement scope JSON. Optional – auto-discovered (see §06). Default-deny if none found.
<code>-n, --names "A,B"</code>	Comma-separated known names → seeds candidate identities (e.g. <code>"Jane Doe,John Roe"</code>).
<code>-f, --fixtures DIR</code>	Offline fixtures dir (defaults to <code>./fixtures</code> if present) – <code><target>.json</code> sample data.
<code>-F, --fixture-file F</code>	An explicit fixture JSON file.
<code>-e, --exposure FILE</code>	Exposure index → flags identities whose email/username is exposed (credential-reuse), boosts their routes.
<code>--no-enrich</code>	Skip enrichment (credential-reuse flags).
<code>-O, --online</code>	Enable networked collectors: live DNS + TLS surface discovery on in-scope hosts.
<code>-P, --passive</code>	Enable third-party passive OSINT (crt.sh subdomain discovery; in-scope domains only).
<code>--timeout SEC</code>	Per-connection timeout for <code>-O</code> probes (default 4).
<code>-c, --collectors L</code>	Comma-separated collector names to run (default: all applicable).
<code>-l, --limit N</code>	Max ranked routes (default 10).
<code>-j, --json</code>	Machine output – NDJSON, one compact object per target (pipes to <code>jq</code>).
<code>--dot</code>	Graphviz DOT of the identity graph (pipe to <code>dot -Tpng</code>).
<code>-q, --quiet</code>	Results only – no banner or decoration.
<code>-v, --verbose</code>	Print pipeline stages to stderr (stdout stays clean for pipes).

HOW RECON WORKS

Five stages, one flow: **Profile** (infer email/username schema) → **Correlate** (fuse usernames/emails/avatars into confidence-scored identities: exact → fuzzy → perceptual avatar-hash) → **Enrich** (credential-reuse) → **Surface** (map in-scope auth surfaces) → **Rank** (identity × surface × evidence → a "most-likely way in" shortlist).

04 RUN / CHECK – FLAGS & MODULES

FLAG	MEANING
<code>-s, --scope FILE</code>	Scope JSON (auto-discovered if omitted).
<code>-o, --option K=V</code>	Set a module option (repeatable), e.g. <code>-o RPORT=8443 -o SSL=false</code> .
<code>--findings FILE</code>	Findings JSONL to append to (default: alongside the scope). <code>--no-findings</code> to skip.

MODULE (ALIAS)	WHAT IT DOES	OPTIONS (DEFAULTS)
<code>tcp_service_probe</code> (probe)	Scope-gated TCP connect + banner grab across common ports.	<code>RHOST</code> , <code>PORTS</code> =21,22,25,80,110,143,443, <code>TIMEOUT</code> =4
<code>http_expose</code> (expose)	Read-only web-exposure audit: <code>.git</code> / <code>.env</code> / <code>.svn</code> , directory listing, status endpoints.	<code>RHOST</code> , <code>RPORT</code> =443, <code>SSL</code> =true, <code>PATHS</code> , <code>TIMEOUT</code> =5
<code>http_fingerprint</code> (fp)	Fingerprint a web surface: server banner, page title, tech tells.	<code>RHOST</code> , <code>RPORT</code> =443, <code>SSL</code> =true, <code>TIMEOUT</code> =5
<code>http_headers</code> (headers)	Security-headers audit: missing HSTS / CSP / X-Frame-Options, version disclosure, insecure cookies.	<code>RHOST</code> , <code>RPORT</code> =443, <code>SSL</code> =true, <code>TIMEOUT</code> =5
<code>tls_audit</code> (tls)	TLS posture: deprecated protocol, weak cipher, expired / self-signed cert.	<code>RHOST</code> , <code>RPORT</code> =443, <code>TIMEOUT</code> =5
<code>ssh_audit</code> (ssh)	SSH banner audit: SSH protocol 1.x, outdated OpenSSH (< 7.4).	<code>RHOST</code> , <code>RPORT</code> =22, <code>TIMEOUT</code> =5

```
$ skirmisher run expose 10.0.0.5           # target → RHOST; records a finding
$ skirmisher run tls 10.0.0.5 -o RPORT=8443 # override an option
$ skirmisher report -o report.md           # then build the report
$ skirmisher run probe 8.8.8.8  x REFUSED # out of scope (default-deny)
```

05 COLLECTORS

COLLECTOR	TIER	WHAT IT DOES
email-schema	offline	Infer likely email/username formats for the target domain.
fixture	offline	Load signals/surfaces from a local JSON fixture (demo/test).
surface-probe	network (-0)	Resolve in-scope hosts + read TLS cert → live auth surfaces.
cert-transparency	passive (-P)	crt.sh subdomain discovery; feeds the active probe.

Offline collectors always run. Networked ones run only under `-0`; third-party ones only under `-P` (and only for in-scope domains).

06 SCOPE – THE SAFETY CORE

```
{
  "engagement": "acme-2026",
  "authorized_by": "Acme CISO",
  "expires": "2026-12-31",
  "allow": ["*.acme.com", "203.0.113.0/24"],
  "deny": ["vpn-legacy.acme.com"]
}
```

ALLOW / DENY ENTRIES domains (`.*` = subdomains only), bare hostnames, IPs, or CIDRs. `engagement` and `authorized_by` are required; an empty allow-list is rejected. **Default-deny**, **deny beats allow**, expiry enforced, every decision audited.

DISCOVERY ORDER `-s FILE` → `./scope.json` → `./scope.example.json` → `$$SKIRMISHER_SCOPE` → `~/config/skirmisher/scope.json`. Exposure & findings files follow the same pattern.

07 OUTPUT & PIPES

```
$ skirmisher acme.com # human report (ranked "way in")
$ skirmisher acme.com -j | jq '.routes[0]' # NDJSON → jq
$ skirmisher acme.com --dot | dot -Tpng -o map.png # the visual identity-surface map
$ cat hosts.txt | skirmisher - -j > out.ndjson # many targets from stdin
$ skirmisher report -o report.md # the pentest deliverable
```

JSON goes to **stdout**, logs/stage timings to **stderr** (**-v**), so output pipes cleanly.

08 EXIT CODES

CODE	MEANING
0	Success.
1	A module check failed / nothing found.
2	Bad usage or error (e.g. no scope, malformed file).
3	Refused — target out of scope.

Scriptable: `skirmisher acme.com && ...`.

09 WORKED EXAMPLE — A FULL ENGAGEMENT

```
$ skirmisher init -g acme.com # once — then edit authorized_by
$ skirmisher acme.com -P -0 # recon: passive + live discovery
$ skirmisher run expose vpn.acme.com # act on a surface (records a finding)
$ skirmisher run headers vpn.acme.com; skirmisher run tls vpn.acme.com; skirmisher run ssh vpn.acme.com
$ skirmisher report -o report.md # the deliverable — a pentest report
```

AUTHORIZED USE ONLY

Skirmisher is for targets you own or are contracted to test. Scope defines legality; the guard enforces it and the audit log records it.