openclawscanner

Lightweight cross-platform detection scripts for **OpenClaw** and its previous names **Moltbot** and **Clawdbot**. When OpenClaw is found, the scripts can also **optionally scan for installed skills** and flag any that match a maintained list of known-malicious skills (currently **341 skills** listed in `risk.txt`, with more to be added over time).
These scripts are designed for:

- **MDM / RMM deployment** (Addigy, Intune, Jamf, JumpCloud, Workspace ONE, Kandji, etc.)
- **Standalone use by incident responders or admins**

Currently the repo ships:

- `detect-openclaw.sh` — macOS / Linux
- `detect-openclaw.ps1` — Windows

Both scripts return machine-readable output and opinionated exit codes for easy policy wiring.

Features

The detectors look for OpenClaw/Moltbot/Clawdbot presence via:

Check	macOS	Linux	Windows
CLI binary (`openclaw`)	Yes	Yes	Yes
CLI version	Yes	Yes	Yes
macOS app (`/Applications/OpenClaw.app`)	Yes	-	-
State directory (`~/.openclaw`, legacy: `~/.moltbot`, `~/.clawdbot`)	Yes	Yes	Yes
Config file (`openclaw.json` / legacy configs)	Yes	Yes	Yes
Gateway service (launchd/systemd/schtasks)	Yes	Yes	Yes
Gateway port (default `18789` plus any ports found in configs)	Yes	Yes	Yes
Docker containers	Yes	Yes	Yes
Docker images	Yes	Yes	Yes

Exit codes

Both scripts use the same basic exit code contract so your MDM policy logic is consistent, with an **optional** extra code when you enable skill scanning:

Exit code	Meaning	MDM status suggestion

Exit code	Meaning	MDM status suggestion
`0`	Not installed / no indicators found	**Success (clean)**
`1`	Installed (running or not), no malicious skills detected	**Error / non-compliant (found)**
`2`	Script error (permissions, environment, etc.)	**Error (investigate)**
`3`	Installed and at least one **malicious skill** (from `risk.txt`) detected — only possible when skill scanning is enabled	**Error / high severity (malicious skill)**

Usage

macOS / Linux

Local run (no skill scan, just core OpenClaw detection):

```
bash detect-openclaw.sh
```

Scan all local users (requires root):

```
sudo bash detect-openclaw.sh
```

Enable **skill scanning** (enumerate installed skills and match against `risk.txt`):

```
bash detect-openclaw.sh --scan-skills
```

Or:

```
sudo bash detect-openclaw.sh --scan-skills
```

When `--scan-skills` is used and OpenClaw is installed, the script will add fields like:

- `skills-installed-count: N`
- `installed-skill: <name> (path: <path/to/SKILL.md>)`
- `malicious-skills-count: M`
- `malicious-skill: <name> (path: <path/to/SKILL.md>)`

and will exit with code `3` if `M > 0`.

You can deploy the script via MDM custom scripts / extension attributes and branch on the exit code and/or parse the `summary:` line and the `malicious-*` fields.

Run **directly from GitHub** (no clone) using the raw script URL:

```
bash <(curl -fsSL https://raw.githubusercontent.com/ibrahimsaleem/openclawscanner/main/detect-openclaw.sh)
```

Windows (PowerShell)

Local run from an elevated or standard PowerShell session (no skill scan, just core detection):

```
powershell -ExecutionPolicy Bypass -File .\detect-openclaw.ps1
```

Enable **skill scanning**:

```
powershell -ExecutionPolicy Bypass -File .\detect-openclaw.ps1 -ScanSkills
```

As with bash, when `-ScanSkills` is used and OpenClaw is installed, the script will emit:

- `skills-installed-count: N`
- `installed-skill: <name> (path: <path/to/SKILL.md>)`
- `malicious-skills-count: M`
- `malicious-skill: <name> (path: <path/to/SKILL.md>)`

and exit with `3` if at least one malicious skill is found.

Or run **directly from GitHub** (no clone) using the raw script URL (you can append `-ScanSkills` if desired):

```
irm https://raw.githubusercontent.com/ibrahimsaleem/openclawscanner/main/detect-openclaw.ps1 | powershell -ExecutionPolicy Bypass -NoProfile -
```

As with the bash variant, you can:

- key off the **exit code** in your MDM/RMM
- parse the **text output** for `summary:`, `skills-installed-count:`, and `malicious-skills-count:` / `malicious-skill:` fields

If you host these scripts on your own HTTP server or Git repository, update any MDM-side URLs to point at your hosted copies.

Environment variables

Both scripts support the same tuning knobs:

Variable	Default	Description
`OPENCLAW_PROFILE`	(empty)	Profile name for multi-instance setups; affects state/config paths and service names
`OPENCLAW_GATEWAY_PORT`	`18789`	Base gateway port to check; additional ports are discovered from config files

Example output

Example from a macOS host with OpenClaw running:

```
summary: installed-and-running
platform: darwin
cli: /usr/local/bin/openclaw
cli-version: 2026.1.15
app: /Applications/OpenClaw.app
state-dir: /Users/alice/.openclaw
config: /Users/alice/.openclaw/openclaw.json
gateway-service: gui/501/bot.molt.gateway
gateway-port: 18789
docker-container: not-found
docker-image: not-found
```

`summary` plus the exit code are usually all you need for compliance / detection policies.

Roadmap

- Further hardening of the skill scanner (additional heuristics beyond the static `risk.txt` malicious list).
- Expand and refresh the malicious skill list in `risk.txt` as new harmful or suspicious skills are discovered. If you identify additional malicious skills, **please open a PR adding them to `risk.txt`** so the wider community benefits from updated detections.

MDM integration guides

Platform-specific deployment notes live under `docs/`:

Platform	Guide
Addigy	`docs/addigy.md`
JumpCloud	`docs/jumpcloud.md`
Microsoft Intune	`docs/intune.md`
Jamf Pro	`docs/jamf.md`
VMware Workspace ONE	`docs/workspace-one.md`

Platform	Guide
Kandji	`docs/kandji.md`

Further reading

For a full technical deep dive (architecture, flow diagrams, and integration model), see:

- `docs/doc.md`

Lightweight detection scripts for macOS, Linux, and Windows that check for CLI binaries, app bundles, config files, gateway services, and Docker artifacts for OpenClaw and its previous names Moltbot and Clawdbot. Designed for MDM deployment or standalone use.

TL;DR

Detection scripts for MDM deployment to identify OpenClaw/Moltbot/Clawdbot installations on managed devices.

What It Detects

Check	macOS	Linux	Windows
CLI binary (`openclaw`)	Yes	Yes	Yes
CLI version	Yes	Yes	Yes
macOS app (`/Applications/OpenClaw.app`)	Yes	-	-
State directory (`~/.openclaw`)	Yes	Yes	Yes
Config file (`~/.openclaw/openclaw.json`)	Yes	Yes	Yes
Gateway service (launchd/systemd/schtasks)	Yes	Yes	Yes
Gateway port (default 18789)	Yes	Yes	Yes
Docker containers	Yes	Yes	Yes
Docker images	Yes	Yes	Yes

Exit Codes

Exit Code	Meaning	MDM Status
0	NOT installed	Success (clean)
1	Installed (running or not)	Error (found)
2	Script error	Error (investigate)

Usage

macOS/Linux (local script)

```
bash detect-openclaw.sh
```

To scan all users, run with `sudo`:

```
sudo bash detect-openclaw.sh
```

Windows (PowerShell, local script)

```
powershell -ExecutionPolicy Bypass -File .\detect-openclaw.ps1
```

If you host these scripts on your own HTTP server or Git repository, update the URLs in your MDM integration to point at your copies.

Environment Variables

Variable	Default	Description
`OPENCLAW_PROFILE`	(none)	Profile name for multi-instance setups
`OPENCLAW_GATEWAY_PORT`	18789	Gateway port to check

Example Output

```
summary: installed-and-running
platform: darwin
cli: /usr/local/bin/openclaw
cli-version: 2026.1.15
app: /Applications/OpenClaw.app
state-dir: /Users/alice/.openclaw
config: /Users/alice/.openclaw/openclaw.json
gateway-service: gui/501/bot.molt.gateway
gateway-port: 18789
docker-container: not-found
docker-image: not-found
```

MDM Integration

Platform	Guide
Addigy	docs/addigy.md

Platform	Guide
JumpCloud	docs/jumpcloud.md
Microsoft Intune	docs/intune.md
Jamf Pro	docs/jamf.md
VMware Workspace ONE	docs/workspace-one.md
Kandji	docs/kandji.md