

Target greatwait

No malicious behavior observed

We ran it 1 time, with a normal setup. It contacted api.greatwait.dev, docs.greatwait.dev and registry.npmjs.org. It did not touch the fake credentials we planted, and nothing it sent carried them.

Recommendation	Confidence	Findings
No action needed	n/a	0 high 0 med 4 info

Nothing observed calls for action. This is evidence, not a guarantee.

Scope

Package	greatwait
Source	clawhub 1.0.0
SHA-256	3bebafe0ecb24cbd0e7ab9f9ec8c3f9d83267ce86498123b0fddd595749812e
Files	5
Scanned	26 Oct 2026, 19:19 UTC
Engine	detonator 0.5.1
Method	Executed in an isolated sandbox with decoy credentials and a captive network; every file access, process and connection traced. 1 runs: baseline.
Note	batch:clawhub:2026-10-26

Findings summary

ID	Severity	Confidence	Finding
F01	Low	Info	Used the planted login with its own service (registry.npmjs.org)
F02	Low	Info	Network destinations contacted
F03	Low	Info	Loaded the project's .env file
F04	Low	Info	A tool read its own credentials (e.g. gh, aws, ssh)

Observed behavior

Install step	package manager only, no install scripts
Runs as	ls -la; ls -la skill; which greatwait gw; npm ls -g --depth=0; node /usr/bin/npm ls -g --depth=0; head -20



Network	api.greatwait.dev (2 conn, 240 B sent; while running) docs.greatwait.dev (3 conn, 271 B sent; while running) registry.npmjs.org (5 conn, 2038 B sent; while running)
Credentials	./env (read) ~/.npmrc (own tool, normal)
Files written	nothing outside its own folder

Finding details

F01 Used the planted login with its own service (registry.npmjs.org)

Severity: Low Confidence: Info Observed in: llm:baseline

A planted login went to the service it belongs to, as normal authentication (npm sending its token to registry.npmjs.org while installing).

Description

A planted login was sent to the service it belongs to, only as the Authorization header of ordinary requests: for example npm sending the ~/.npmrc token to registry.npmjs.org while installing dependencies.

Impact

None in itself: this is how the package manager authenticates. Publishing, unpublishing or creating tokens with it would be reported as exfiltration.

Evidence

```
~/.npmrc (npm) sent only as the login header of ordinary requests (no publish)
```

Remediation

No action needed.

F02 Network destinations contacted

Severity: Low Confidence: Info Observed in: llm:baseline

The package connected to these hosts.

Description

Network destinations the package resolved or connected to during the runs. All traffic was captured by the sandbox; nothing reached the internet.

Impact

Context only. Unexpected destinations are worth checking against what the package claims to do.

Evidence

```
api.greatwait.dev
```

```
docs.greatwait.dev
```

```
registry.npmjs.org
```

Remediation

Confirm each host is expected (the package's API, its registry). Allowlist expected egress in production.



F03 Loaded the project's .env file

Severity: Low Confidence: Info Observed in: llm:baseline
The package loaded the project's .env file.

Description

The process loaded the project's .env file. Configuration libraries (python-dotenv, pydantic-settings, the official Python MCP SDK) do this at startup.

Impact

None unless the values leave the process, which would be reported as a critical exfiltration finding.

Evidence

```
./.env (by sed)
```

Remediation

No action needed. Keep long-lived secrets out of project .env files where possible.

F04 A tool read its own credentials (e.g. gh, aws, ssh)

Severity: Low Confidence: Info Observed in: llm:baseline
A tool opened its own login or config file, such as npm reading ~/.npmrc.

Description

A tool opened its own configuration or login file, such as npm reading ~/.npmrc or gh reading its hosts file.

Impact

None in itself: this is how those tools work. Listed so the report shows every credential access.

Evidence

```
~/.npmrc (by npm)
```

Remediation

No action needed.

Run log

llm:baseline ok 63.4s

new behavior: dns:api.greatwait.dev, dns:docs.greatwait.dev, dns:registry.npmjs.org, exec:curl, ownread:./.env | 8 model steps, 8 commands

Observed behavior only. A clean result does not prove a package is safe. Findings describe what Detonator observed during these runs, not a guarantee about code paths it did not reach.