

Teaching AI to Fix Container Vulnerabilities

Anton Sankov

Anton Sankov

Staff AI Engineer at Cast AI

- Doing Container Security stuff since 2020
- Currently working on AI agents

 Anton Sankov

 asankov.dev



Agenda

1. What is an AI Agent
2. Attempt #1
3. Lessons Learned and Attempt #2
4. Will the AI Replace Us

DISCLAIMER:

This talk covers a commercial non-open source product.

This is not a sales pitch, it's just a knowledge and experience sharing session.

AI AGENTS



SO HOT RIGHT NOW

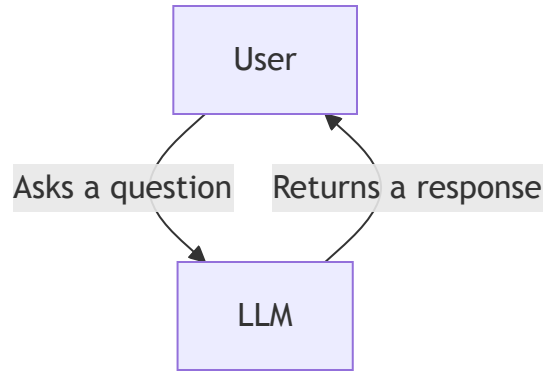
What is an AI Agent?

AI agents are software systems that use AI to pursue goals and complete tasks on behalf of users.

They show reasoning, planning, and memory and have a level of autonomy to make decisions, learn, and adapt.

Chat vs Agent

Chat

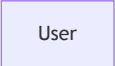


Execution time: 1 - 30 seconds

Chat vs Agent

Agent

Execution time: 5 - 60 minutes



The Problem

Endless lists of vulnerabilities

The Solution

Build an AI agent that remediates security vulnerabilities (CVEs) in container images.

Input:

- container image
- git repository (where the source of the container image is)
- a list of vulnerabilities inside the container image
 - system (Linux packages, binaries, etc.)
 - application (Java/JS/Python/Go, etc. libraries and dependencies)

Expected output:

- a Pull Request with changes that remediate vulnerabilities in that image

Attempt #1

Use multiple AI agents, but orchestrate everything in code

```
1  for image in vulnerable_images:
2      # an AI agent that finds the Dockerfile in the code repo
3      dockerfile = find_dockerfile(image.git_repo)
4
5      # an AI agent that remediates vulnerabilities in system(OS) packages
6      remediate_system_vulnerabilities(image.vulnerabilities, dockerfile)
7
8      # an AI agent that finds the application dependencies file (package.json, pom.xml, build.gradle, etc.)
9      dependency_file = find_dep_file(image.git_repo)
10
11     # an AI agent that remediates vulnerabilities in the application dependencies
12     remediate_application_vulnerabilities(image.vulnerabilities, dependency_file)
13
14     # an AI agent that finds the build command and builds the image with the new changes
15     new_image = build_image(image.git_repo)
16
17     # [deterministic] scan the newly build image
18     scan_results = scan_image(new_image)
19
20     # MORE DETERMINISTIC LOGIC – compare results, commit changes, open a PR, etc. ...
```

Attempt #1

Fallacies of that approach

Each one of these agents we invoke is a potential point of failure. Upon failure of one agent, we failed the whole thing. Probability of failure is:

$$1 - (1 - M)^N$$

where:

M probability of failure of 1 agent

N number of agents

Attempt #1

Fallacies of that approach

**Sometimes the build would fail,
because of lack of resources
(looking at you, Java apps)**

Attempt #1

Fallacies of that approach

The agent would produce changes for images that the users don't care about

Attempt #1

Fallacies of that approach

**Sometimes the agent would just produce
nonsense**

Attempt #1

Nonsense like this:

```
31 + # ---- Patch CVE: upgrade bundled pip (24.0 → 24.1.2) ----
32 + # Note: The bundled Python in Google Cloud SDK is glibc-linked and may not run on Alpine (musl).
33 + # To ensure the bundled pip is upgraded without executing the bundled interpreter, we fetch
34 + # the pip 24.1.2 wheel and extract it directly into the bundled site-packages directory.
35 + RUN set -eux; \
36 +     PIP_VERSION=24.1.2; \
37 +     SITE_PKGS=/google-cloud-sdk/platform/bundledpythonunix/lib/python3.11/site-packages; \
38 +     mkdir -p "${SITE_PKGS}"; \
39 +     PIP_WHL_URL=$(curl -s "https://pypi.org/pypi/pip/${PIP_VERSION}/json" | jq -r '.urls[] | select(.filename=="pip-""${PIP_VERSION}""-py3-none-
any.whl") | .url'); \
40 +     wget -O /tmp/pip-${PIP_VERSION}-py3-none-any.whl "${PIP_WHL_URL}"; \
41 +     python3 - <<'PY'
42 + import zipfile, os, glob, shutil, pathlib, subprocess, sys
43 +
44 + wheel = '/tmp/pip-24.1.2-py3-none-any.whl'
45 + dest = '/google-cloud-sdk/platform/bundledpythonunix/lib/python3.11/site-packages'
46 +
47 + # Extract the new pip wheel into site-packages to replace runtime modules
48 + with zipfile.ZipFile(wheel) as z:
49 +     z.extractall(dest)
50 +
51 + # Also replace any bundled wheels used by ensurepip in both the runtime copy and
52 + # the Google Cloud SDK installer backup directory reported by scanners
53 + ensurepip_dir = pathlib.Path('/google-cloud-sdk/platform/bundledpythonunix/lib/python3.11/ensurepip/_bundled')
54 + for f in ensurepip_dir.glob('pip-24.1-py3-none-any.whl'):
55 +     f.unlink()
56 + shutil.copy('/tmp/pip-24.1.2-py3-none-any.whl', ensurepip_dir / 'pip-24.1.2-py3-none-any.whl')
57 +
58 + backup_dir = pathlib.Path('/google-cloud-sdk/.install/.backup/platform/bundledpythonunix/lib/python3.11/ensurepip/_bundled')
59 + if backup_dir.exists():
60 +     for f in backup_dir.glob('pip-24.1-py3-none-any.whl'):
61 +         f.unlink()
62 +     shutil.copy('/tmp/pip-24.1.2-py3-none-any.whl', backup_dir / 'pip-24.1.2-py3-none-any.whl')
63 +
64 + # Cleanup downloaded wheel
65 + os.remove(wheel)
66 + PY
67 + # -----
68 +
```

Attempt #2

Consolidate logic into a self-orchestrating single AI agent with multiple tools

```
1  for image in vulnerable_images:
2      result = agent.run(prompt=f"""
3      You are a security engineer tasked with remediating security vulnerabilities in a container image.
4
5      You are given a git repo that contains the source code of a container image: {image_name}
6      You are tasked with fixing the following vulnerabilities: {vulnerabilities}
7
8      Use the provided tools to explore the repo and find relevant files
9      (Dockerfiles, application dependency files, etc.)
10
11     Once you are done fixing, try to build the image to make sure you haven't broken anything.
12     If building fails because of your changes, adapt them until the build is successful.
13     If building fails for unrelated reasons, continue without building.
14
15     If you're able to build the image, scan it to make sure the vulnerabilities are fixed.
16 """, tools=[
17     read_file_tool,
18     build_tool,
19     scan_image_tool,
20     open_pr_tool, ... # and more tools
21 ], context={
22     'image_name': image.name,
```

Attempt #2

Gave more control to the users

Vulnerabilities Packages

Q Enter search keywords

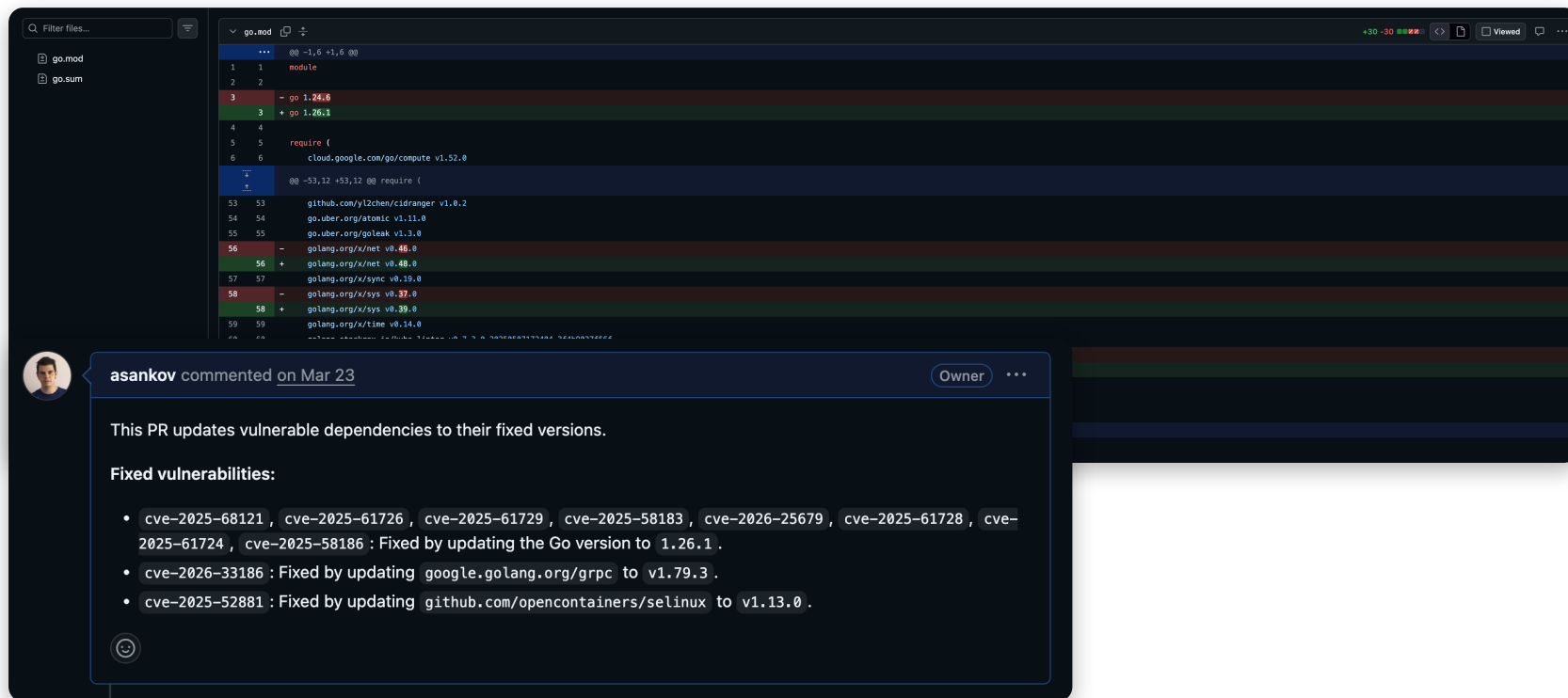
Vulnerability: Active Fix: All Pkg. type Severity Clear all

985 vulnerabilities

<input type="checkbox"/>	CVE ↑↓	SCORE ↓	FOUND IN PACKAGE ↑↓	FIX ↑↓	FIRST DETECTED ↓	
<input type="checkbox"/>	CVE-2025-68121 crypto/tls: crypto/tls: Incorrect certificate validation during TLS session resumption	10	gobinary/stalib v1.22.2	1.24.13, 1.25.7, 1...	a day ago	Fix
<input type="checkbox"/>	CVE-2024-41110 moby: Authz zero length regression	10	gobinary/github.com/docke r/docker v24.0.7+incompati...	23.0.15, 26.1.5, ...	2 months ago	Fix
<input type="checkbox"/>	CVE-2025-68121 crypto/tls: crypto/tls: Incorrect certificate validation during TLS session resumption	10	gobinary/stalib v1.21.5	1.24.13, 1.25.7, 1...	a day ago	Fix
<input type="checkbox"/>	CVE-2025-68121 crypto/tls: crypto/tls: Incorrect certificate validation during TLS session resumption	10	gobinary/stalib v1.22.3	1.24.13, 1.25.7, 1...	a day ago	Fix
<input type="checkbox"/>	CVE-2025-68121 crypto/tls: crypto/tls: Incorrect certificate validation during TLS session resumption	10	gobinary/stalib v1.21.3	1.24.13, 1.25.7, 1...	a day ago	Fix
<input type="checkbox"/>	CVE-2025-68121 crypto/tls: crypto/tls: Incorrect certificate validation during TLS session resumption	10	gobinary/stalib v1.22.3	1.24.13, 1.25.7, 1...	a day ago	Fix
<input type="checkbox"/>	CVE-2025-68121 crypto/tls: crypto/tls: Incorrect certificate validation during TLS session resumption	10	gobinary/stalib v1.22.5	1.24.13, 1.25.7, 1...	a day ago	Fix
<input type="checkbox"/>	CVE-2025-68121 crypto/tls: crypto/tls: Incorrect certificate validation during TLS session resumption	10	gobinary/stalib v1.22.5	1.24.13, 1.25.7, 1...	a day ago	Fix

The Result

Small, boring ... but mergeable PRs



The screenshot displays a GitHub pull request interface. The top section shows a diff for the `go.mod` file. The diff highlights several changes to dependencies, with some updates marked as fixes for vulnerabilities. The changes include:

- Updating `go` from `1.24.0` to `1.26.1`.
- Updating `cloud.google.com/go/compute` from `v1.52.0` to `v1.53.12`.
- Updating `github.com/y12chen/cidranger` from `v1.0.2` to `v1.1.0`.
- Updating `go.uber.org/atomic` from `v1.11.0` to `v1.3.0`.
- Updating `go.uber.org/goleak` from `v1.3.0` to `v1.3.0`.
- Updating `golang.org/x/net` from `v0.46.0` to `v0.48.0`.
- Updating `golang.org/x/sys` from `v0.37.0` to `v0.39.0`.
- Updating `golang.org/x/time` from `v0.14.0` to `v0.14.0`.

Below the diff, a comment from `asankov` (Owner) is visible, dated `Mar 23`. The comment reads:

This PR updates vulnerable dependencies to their fixed versions.

Fixed vulnerabilities:

- `cve-2025-68121`, `cve-2025-61726`, `cve-2025-61729`, `cve-2025-58183`, `cve-2026-25679`, `cve-2025-61728`, `cve-2025-61724`, `cve-2025-58186` : Fixed by updating the Go version to `1.26.1` .
- `cve-2026-33186` : Fixed by updating `google.golang.org/grpc` to `v1.79.3` .
- `cve-2025-52881` : Fixed by updating `github.com/opencontainers/selinux` to `v1.13.0` .

Attempt #2

Why this worked better

Mistake #1: LLMs tend to overdo things, especially when given tasks that are not properly defined and scoped

Solution: Define the task better and limit the context given to the LLM

Mistake #2: We were doing stuff which was not mandatory and could have been done on a best-effort basis

Solution: Recover from failures such as not being able to find a file or a failing build and continue executing the task

Mistake #3: We were treating all images as equal

Solution: Give the user control over which images to fix

Attempt #2

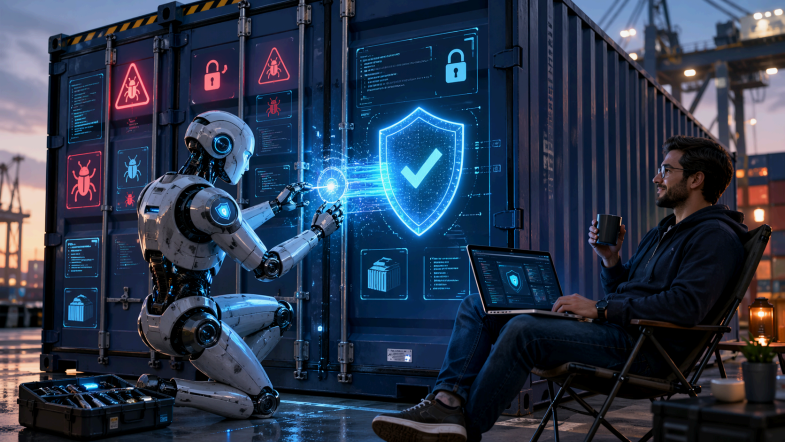
Lessons Learned

LLMs overthink and overdo stuff, but they can be prompted into doing the right thing.

LLMs are resourceful and can find their way out of a problem if given enough freedom.



This problem is as much technical as it is organizational.

Will AI Replace Us?



Thank you!

Questions?

 Anton Sankov
 asankov.dev

Download the slides



asankov.dev/teaching-ai-to-fix-vulnerabilities