[un]prompted II · Workshop

Harness Engineering for Vulnerability Research

aka how 2 hack good, or something

Workbook:
https://github.com/grahamhelton/harness-engineering-workshop-book

Graham Helton · Ryan Basden

San Francisco · October 2026



Email contact@grahamhelton.com
Web grahamhelton.com
Signal [removed]
LinkedIn grahamhelton

Graham Helton

Senior Red Team Engineer, Snowflake
Founder, Low Orbit Security

Kubernetes, Linux, and cloud infrastructure



Email contact@ryanbasden.com
Web ryanbasden.com
Signal [removed]
LinkedIn ryan-basden

Ryan Basden

Head of Offensive Security, Xora
Founder, Empiric Security

Vulnerability research and autonomous source code security testing

AGENDA

How this workshop works

Introductions → Problems → **Harness components, one at a time**

Repeat for each harness component



1

We talk about it

2

We show it conceptually with a visual

3

A quick demo, if needed

4 · You build · ~10 minutes

Implement it in your harness

This is hard and will require brain power. This is your time to ask questions and get ideas.

This is the most important part of the workshop.

THE PROBLEM

“Find all the vulnerabilities in this repo”

- Unsupported claims
- No controlled input or output
- No demonstrated reachability
- No runtime validation
- Unusual behavior treated as security impact
- No useful reproduction or disclosure material

Model capability is not research capability.

A capable model can still produce weak vulnerability research when it lacks target context, deterministic tools, authorization boundaries, durable state, and a validation method.

Problems in Agent-Assisted Vulnerability Research

An agent may report the app started when only a container was created. 01 → **Local Validation**	A local checkout alone cannot establish current GitHub settings. 02 → **External Integrations**	Without an interface, each interaction starts with a small discovery task. 03 → **Operational Interfaces**
Compaction loses key details; observations are lost when the session closes. 04 → **Memory**	Too little information and it cannot assess the question; too much and irrelevant material can influence its answer. 05 → **Context Management**	Dumping too much telemetry into the context window quickly overloads it. 06 → **Telemetry**
You keep instructing a model how to do the same task. 07 → **Skills**	A scanner match does not establish attacker control, reachability or impact. 08 → **Deterministic Tools**	Relying on the model to follow instructions is not sufficient. 09 → **Hooks**



Tailworm, Oct 2026 | https://web.archive.org/web/20261026090039/https://tailworm.com/

CASE STUDY

onn4a submitted a report to curl. September 15, 2026, 6:30pm UTC

Summary:

During a file upload, curl mishandles HTTP/2 RST_STREAM received from the server (with reason HTTP_1_1_REQUIRED or REFUSED_STREAM) leading to a use after free of a `struct Curl_easy` object. The control flow is a bit complicated, so I've produced an ASCII tree of the control flow to highlight the object lifetime issue in the Appendix below.

Preconditions for this are `curl -T <file>`, where `<file>` contains anywhere "<word> <word> <any>\n\n" and exceeds ~135KB. Our PoC exploit also requires `-Z` / `--parallel` when PIE/ASLR is enabled, but this is not necessary to trigger the memory corruption, only for exploitation.

As for how the UAF happens, after receiving the reset, multi.c converts the reset into a retry and calls `multi_done()`. In `multi_done()`, it calls `Curl_conn_ev_data_done()` which makes curl forget about the transfer, then it calls `Curl_req_done()`. Then `Curl_req_done()` flushes the buffered unsent remainder of the upload body into the connection. Since curl no longer knows about the transfer, it treats those bytes as a new request. If they look like a request line, `h2_submit()` opens a new stream that stores a pointer to the `Curl_easy` object. This is the pointer that will be UAFed. After the retry completes, the application frees the `Curl_easy` object, but the connection still holds the pointer. It is dereferenced when the server sends a frame on that stream which can happen at exit, when the connection pool is destroyed.



bagder curl staff posted a comment.

I must say @onn4a_luna that your report, the reproducer and the level of details are all excellent.

Curl, Oct 2026 · hackerone.com/reports/4030333

BEFORE WE BUILD

Pick a target

You'll research one real open-source project for the rest of the workshop.

WHERE TO LOOK

Awesome Self-Hosted

a directory of self-hostable open-source apps

awesome-selfhosted.net

CNCF Landscape

cloud-native projects, filter by category

landscape.cncf.io

You have to be able to **run it locally**.

01

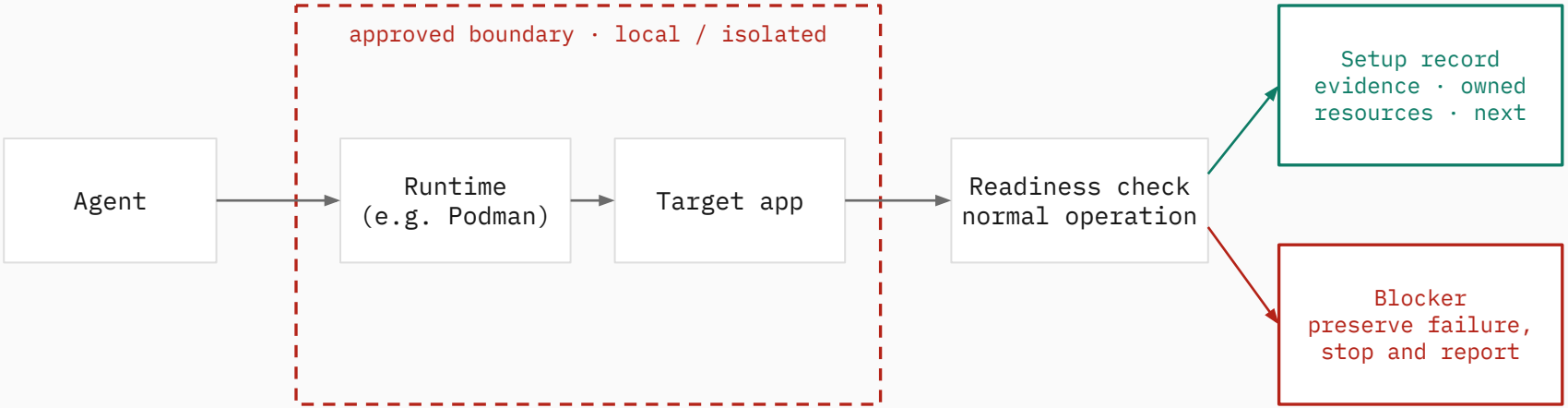
Local Validation Environment

A known way to run the target application and the tools the agent needs.

Local Validation Environment

- “Started” is not “working”: a container is not a running app
- Define readiness: app reachable, tools functional, a harmless normal operation succeeds
- Give the agent standards it can check (AGENTS.md: runtime, tools, boundary, stop conditions)
- Instructions describe the boundary; isolation enforces it
- Checkpoint: a setup record, or a precise blocker

Setup is done when there is evidence



"command -v podman curl" finds paths. It does not show the app works.

“Started” is not “working”

1

`command -v podman curl`

Shows
Command paths exist

Does not show
Engine reachable

2

`podman ps`

Shows
A container exists

Does not show
The app is running

3

`curl 127.0.0.1:13173`

Shows
Something answers

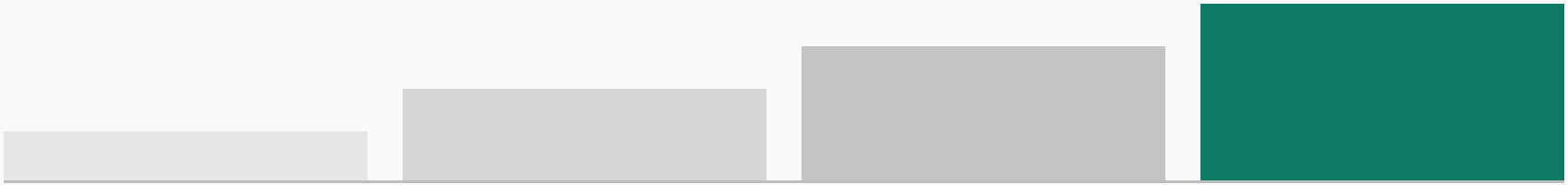
Does not show
That it is Gitea

4

`curl …/api/healthz`

Shows
Gitea responds normally

Result
Ready



Its report should point to observed output, not just a statement that setup is complete.

01 · YOUR TURN

Set up a local validation environment

Build

1. Start from the project's normal setup
2. Have the agent run the target inside your isolated environment
3. Add a readiness check that proves normal operation

Done when

The agent's setup report points to observed output, not a statement that setup is complete.

Smallest version

The target running locally and one request that returns a normal response.

Going further

Access control
Prepare two users, an administrator, and resources with known owners.

Several services
Add database migrations, a queue, and a worker.

Nix flake
Define the compiler, runtime, and research tools with locked dependency versions.

02

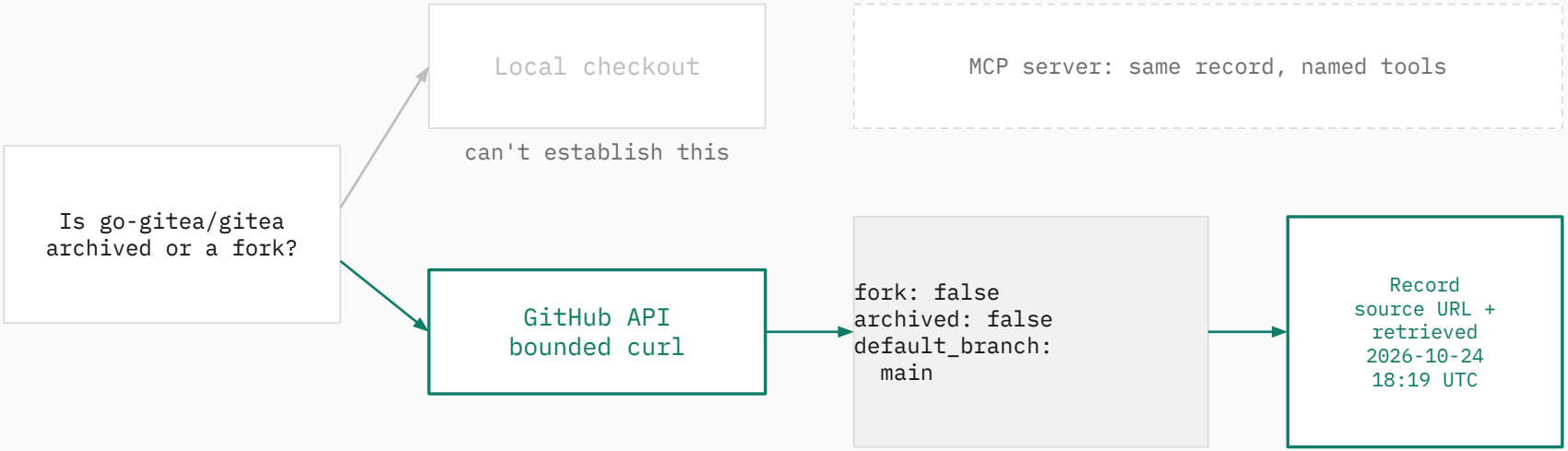
External Integrations

Access to information or actions outside the agent's local workspace.

External Integrations

- Start with a question the local checkout can't answer
- Go to the system that maintains the record (GitHub API, advisories)
- Bounded requests: timeouts, size limits, no credentials by default
- Keep source URL and retrieval time: a fact, not prior knowledge
- MCP adds prepared tools, not a more authoritative record

One question, one bounded request, one record



Response text is untrusted data. It cannot change the task or grant permission.

Go to the system that maintains the record

Question

Is go-gitea/gitea archived or a fork, and what is its default branch?

Bounded request

`GET /repos/go-gitea/gitea`

```
https only
15 s max
256 KB max
no credentials
no retries · no redirects
```

Response check identity fields first

```
{
  "full_name": "go-gitea/gitea",
  "private": false,
  "fork": false,
  "archived": false,
  "default_branch": "main"
}
```

Local checkout
cannot establish current GitHub settings

Record

source `https://api.github.com/repos/go-gitea/gitea` retrieved `2026-10-24 18:19:19 UTC`

The source URL and retrieval time connect the values to a particular record.

02 · YOUR TURN

Add an external integration

Build

1. Write one question the local checkout cannot answer
2. Make one request to the system that maintains the record
3. Save the response with its source

Done when

You can point to the record that answers the question and where it came from.

Smallest version

One request to the GitHub API for your repository.

Going further

Playwright MCP
Browser actions and structured page snapshots.

GitHub advisory API · OSV
Records to compare with the target's version and code.

disclose.io lookup
Who likely controls an asset, and where a report should go.

03

Operational Interfaces

A clear way to perform an action or inspect the work.

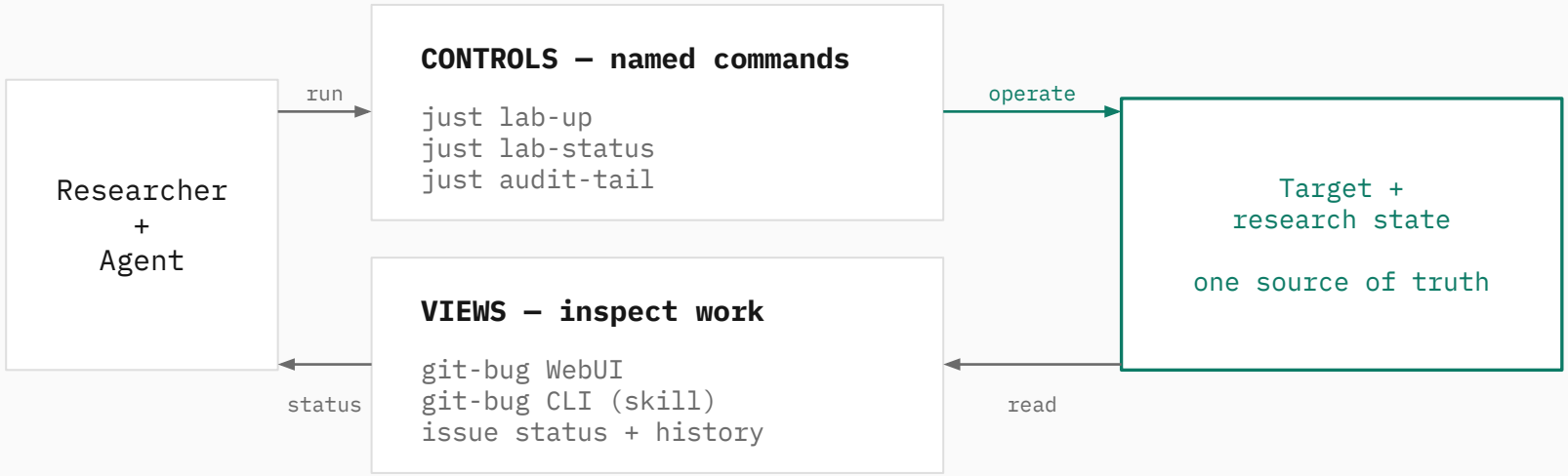
Operational Interfaces

- A clear way to perform an action or inspect the work.
- **Named commands**: a consistent way to perform tasks.
- **Views**: helps you understand what is happening.
- Different interfaces must operate on the same underlying data.
- Build an interface for what you repeatedly reconstruct or struggle to find.

```
research-lab

$ just lab-up
$ just lab-status
$ just audit-tail
```

Controls and views



When people and agents use different interfaces, make sure they operate on the same underlying data.

Which command starts the target?

Request

start the instance

$ **just --list**

```
Available recipes:
  creds  # create the local trainer account if needed, …
  info   # show the recorded instance, source and image …
  logs   # follow the recorded owned container's logs; …
  setup  # preload pinned Gitea source into this lab's …
  start  # start an owned Gitea container on loopback …
  stop   # stop only the recorded owned container and …
```

Agent

just info
State: exited

just start
State: running · URL: http://127.0.0.1:13172/

just info
State: running

Answer

The lab instance is running at http://127.0.0.1:13172/. just info confirms it is running.

Observed session, 2026-10-25

A good interface should make the intended way to perform a task more visible.

What is the status of current work?

Researcher git-bug WebUI

Audit the single-writer paths of the users database (security coverage gap) #fbec351

Open Graham Helton opened this issue 21 days ago

The security audit of 2026-08-25 stopped at a usage limit before the durability and single-writer check ran. No audit examined these three paths after that date:

Agent git-bug skill + CLI

What is the status of the single-writer paths issue?

Read SKILL.md
`git-bug skill`

gb ls
`fbec351  open  Audit the single-writer paths …`

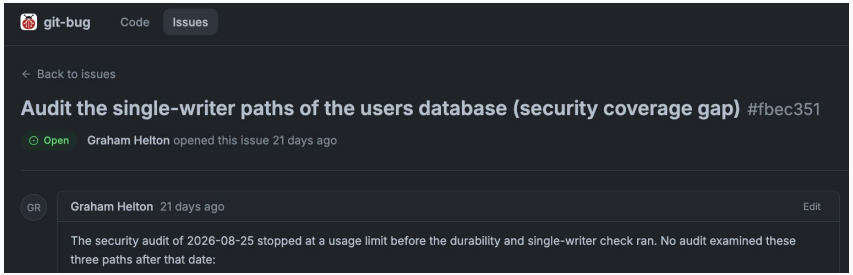
gb show fbec351
`+ 22 lines`

Issue fbec351, “Audit the single-writer paths of the users database”, is open. It has no progress comments; its last edit was October 1.

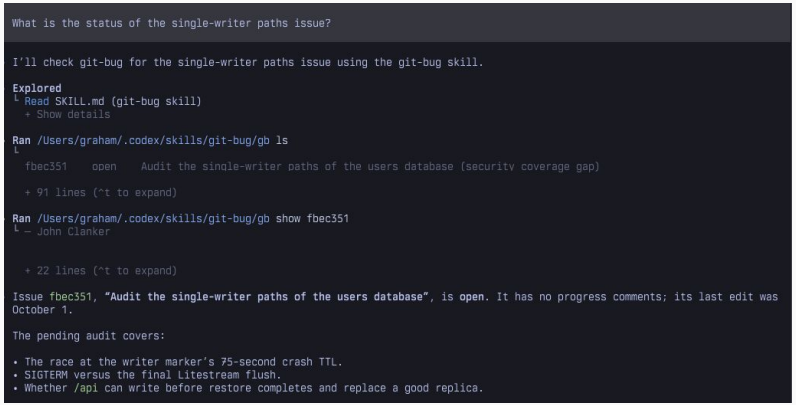
`git-bug issue fbec351` same underlying data

When people and agents use different interfaces, make sure they operate on the same underlying data.

Two interfaces, one issue



Researcher · Git-bug WebUI



Agent · Git-bug skill + CLI

Same issue fbec351, same underlying data.

03 · YOUR TURN

Add an operational interface

Build

1. List the commands you keep retyping or re-explaining
2. Turn two or three into named commands, for example in a justfile
3. Tell the agent about them in AGENTS.md

Done when

A fresh session starts your target without working out how.

Smallest version

One named command that starts your target.

04

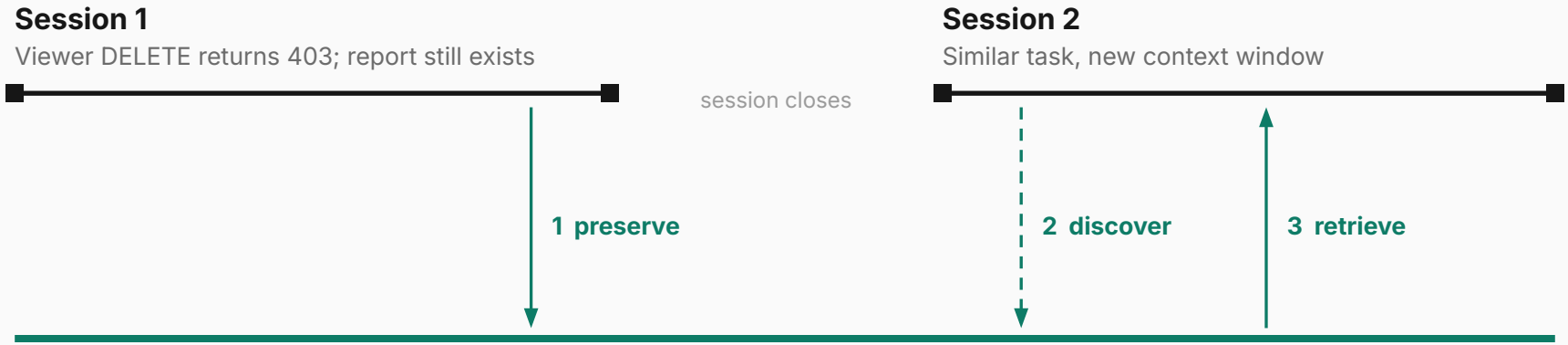
Memory

Carry observations and outcomes into the next agent session.

Memory

- Carry observations and outcomes into the next agent session.
- Compaction and closed sessions lose key details.
- **Preservation**: information survives the current session.
- **Discovery**: a later session can find it.
- **Retrieval**: a later session can access and inspect it.

Preserve, discover, retrieve



`threads/thr-001/` persists across sessions

`index.md`	question · assessment · next action
`evidence/E-001.md`	request · response · state check · limit

What's in a thread

index.md

Where the work currently stands

```
## Question
Can a viewer delete a report they
should not control?

## Current assessment
The tested deletion request did not
delete the report.

## Evidence
- E-001: Viewer deletion attempt

## Next action
Check whether another deletion path
is reachable in the isolated target.
```

A later check may change this assessment.

evidence/E-001.md

What happened in a particular check

```
## Conditions
viewer-a (viewer) · report 42, owned
by author-b · example-revision

## Request
DELETE /api/reports/42

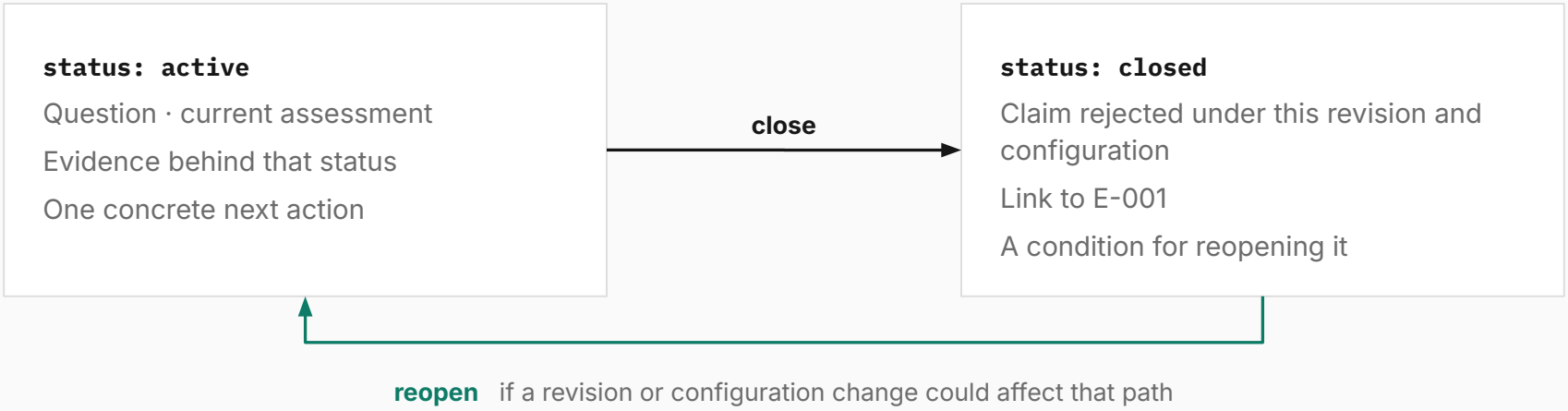
## Observed response
HTTP/1.1 403 Forbidden

## Target-state check
GET as author-b → 200 OK; report present

## Limit
Covers only this path and these conditions.
```

A later check does not change this record.

Thread lifecycle



Closing the tested path does not close the broader viewer-deletion question.

04 · YOUR TURN

Add memory to your harness

Build

1. Create one notes file for your research
2. In AGENTS.md, tell the agent to read it first and record what it tested and what happened
3. Close the session and start a new one

Smallest version

One line in AGENTS.md that points at one notes file.

Done when

The new session can tell you what was tested and what happened.

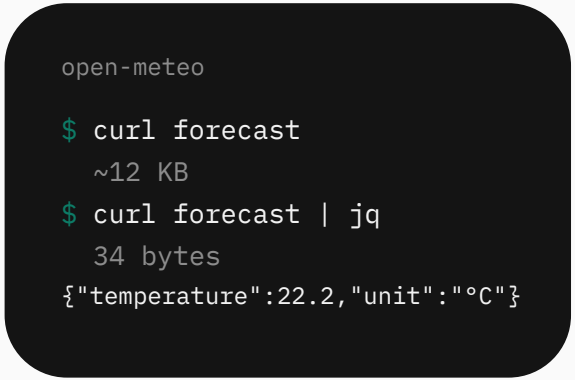
05

Context Management

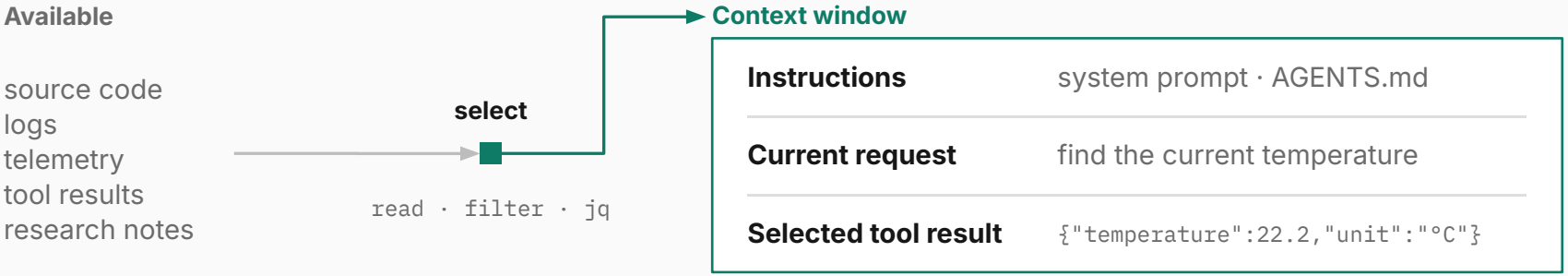
Deciding what information the agent receives for its current task.

Context Management

- Deciding what information an agent receives for a task.
- **Too little**: the agent cannot assess the question.
- **Too much**: irrelevant material can influence its answer.
- A file is not in context until its contents are read.
- Goal: enough relevant information to complete the task.



Select what reaches the context window



Same request, measured Open-Meteo · current temperature in San Francisco

`curl`	~12 KB
`curl | jq`	34 B

Both contained the current temperature and its unit.

Know what you're looking for

$ **kubectl get pods -A -o json**

```
{ "apiVersion": "v1", "kind": "List", "items": [
  { "metadata": {
      "name": "kube-proxy-7xk2p",
      "namespace": "kube-system",
      "uid": "3f9c2a1e-8d4b-4e6a-9b1f-0c7d5e2a4b61",
      "labels": { "k8s-app": "kube-proxy", ... },
      "ownerReferences": [ { "kind": "DaemonSet", ... } ],
      "managedFields": [ ... ] },
    "spec": { "containers": [ { "name": "kube-proxy",
        "image": "registry.k8s.io/kube-proxy:v1.30.2",
        "command": [ "/usr/local/bin/kube-proxy", ... ],
        "securityContext": { "privileged": true },
        "volumeMounts": [ ... ], ... } ],
      "hostNetwork": true, "tolerations": [ ... ], ... },
    "status": { "phase": "Running", "podIP": "10.0.0.4",
      "containerStatuses": [ { "ready": true, ... } ] } },
  { "metadata": { "name": "coredns-5d78c9869d-x2kpl", ... },
    "spec": { "containers": [ { "securityContext": {
        "allowPrivilegeEscalation": false, ... } } ] }, ... },
  ... ] }
```

Question

Which pods run privileged containers?

select

```
jq -r '.items[]
  | select(any(.spec.containers[];
      .securityContext.privileged))
  | "\(.metadata.namespace)  \(.metadata.name)"'
```

Context window

```
kube-system    kube-proxy-7xk2p
monitoring     node-exporter-q9d4f
```

Illustrative output

Some of it is useful, some of it is not. Know what you're looking for.

05 · YOUR TURN

Manage what reaches the context window

Build

1. Pick one command on your target that returns more than you need
2. Write down the one question you want it to answer
3. Filter the output so only the answer reaches the agent

Done when

You can say what question the output answers and what was cut.

Smallest version

One jq or grep filter on one noisy command. Start with logs.

06

Telemetry

What the harness captures about agent and system activity.

Telemetry

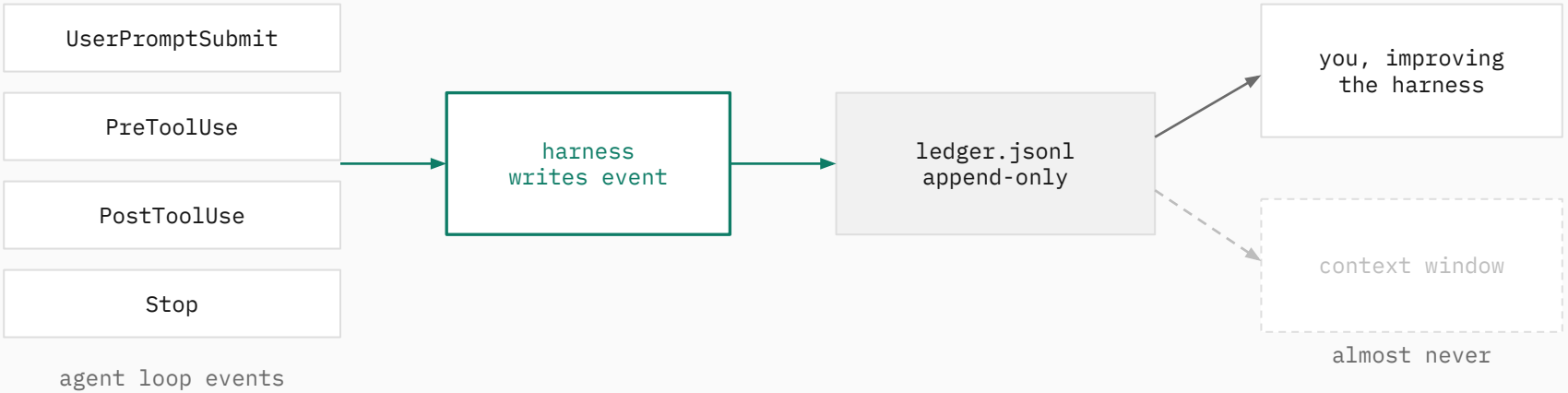
- Recorded by the harness where the event happens, not the agent's own account
- Survives compaction and deleted conversations
- Records activity; doesn't prove a conclusion or impact
- High volume: almost never belongs in the context window
- Decide what to keep, at what detail, for how long, for whom

ledger.jsonl · append-only

```
{"event":"UserPromptSubmit"}
{"event":"PreToolUse","tool":"Bash"}
{"event":"PostToolUse","ms":21}
{"event":"Stop","status":"ok"}
```

one line per event, outside the agent

Recorded outside the agent




```
{"event":"PostToolUse","session_id":"abc123","tool":"Bash","command":"python3 --version","duration_ms":21,"status":"completed"}
```

06 · YOUR TURN

Add telemetry to your harness

Build

1. Decide which events are worth preserving
2. Append one line per event to ledger.jsonl
3. Keep it outside the context window unless a task needs it

Done when

You can answer "what did the agent run?" from the ledger, not the chat.

Smallest version

One line per tool call: the tool and its command.

07

Skills

A reusable set of instructions for a particular kind of work.

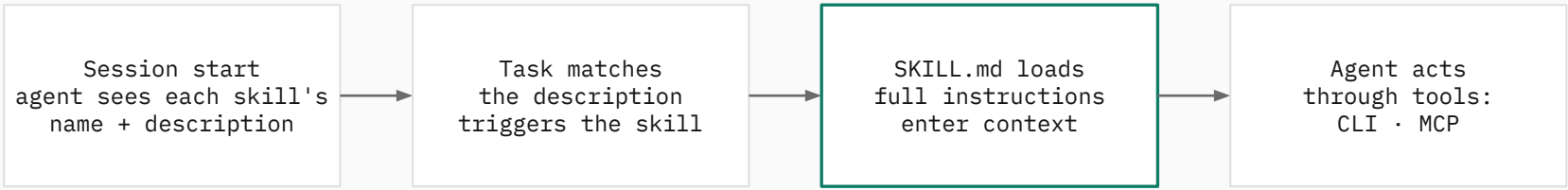
Skills

- A skill is a reusable set of instructions that give an agent additional guidance for a particular kind of work
- They give agents a method for a recurring task instead of requiring the model to reconstruct that method during every session
- Skills improve procedural consistency, but do not provide determinism
- At a high level, a skill is just a folder containing a SKILL.md file
- Tools are still what perform the actions defined in the skill's markdown instructions

SKILL.md

```
---
name: report-research-progress
description: Use when evidence
  changes an investigation's
  assessment or next action.
---
```

How a skill enters the agent loop




```
report-research-progress/
├── SKILL.md
├── references/reporting-tool.md
└── assets/update-template.md
```

Skill loaded at the wrong time?
Refine the description.

07 · YOUR TURN

Write a skill

Build

1	Pick a procedure you instructed the agent on more than once
2	Write a SKILL.md with a name and a description that says when to use it
3	Put the procedure in the body

Done when

It triggers on the right request and not on an unrelated one.

Smallest version

One SKILL.md with its frontmatter and a short list of steps.

08

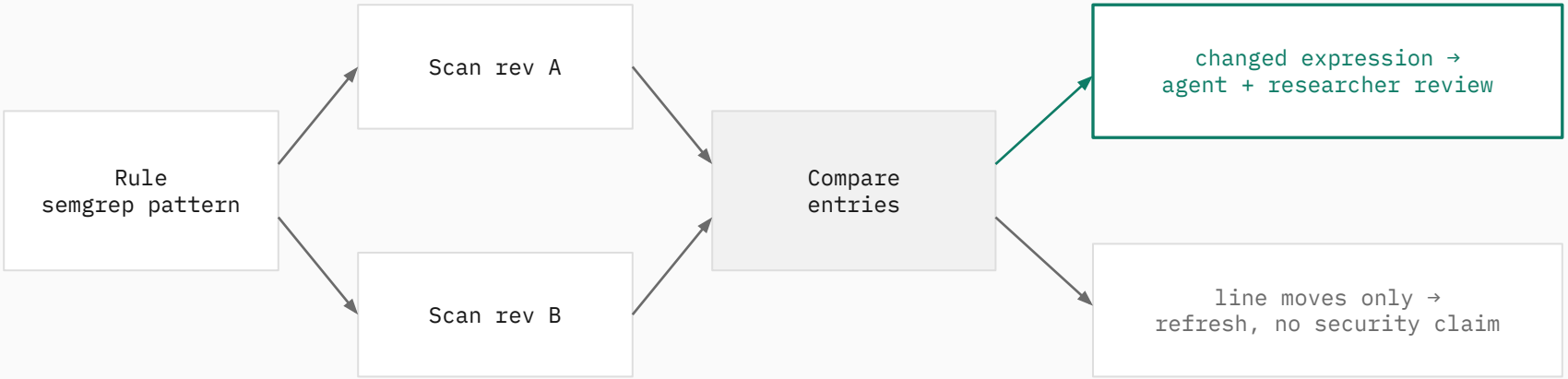
Deterministic Tools

Use a tool for a defined, repeatable question.

Deterministic Tools

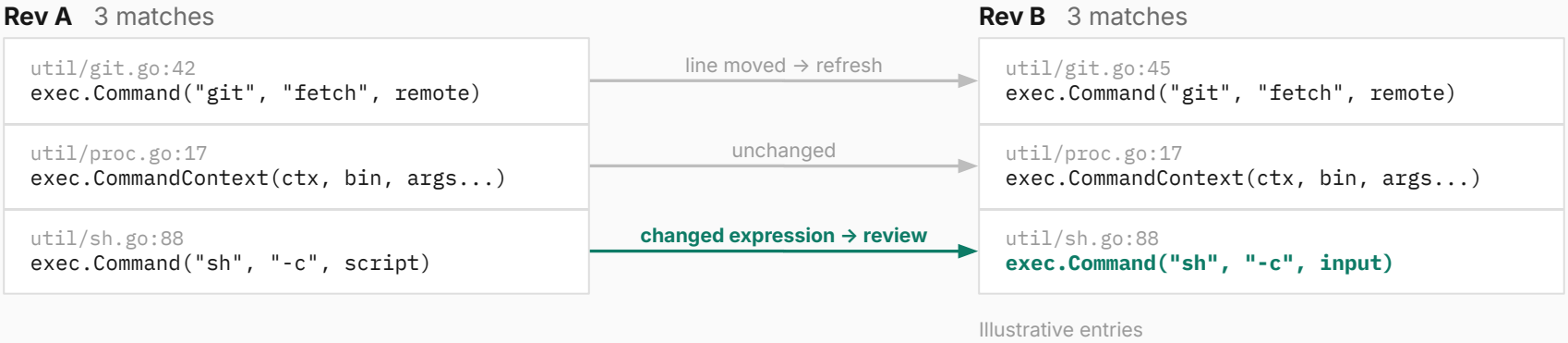
- Don't ask a model to imitate a parser
- Example: inventory exec.Command calls under util/
- Rule + scan + saved comparison → which changes need review
- A match is not attacker control, reachability or impact
- Check coverage: default ignores can silently drop files

Rule, scan, compare, then review



Coverage gap seen in the pilot: default ignores skipped util/test/testutil.go

Same count, different code



Coverage gap seen in the Argo pilot

`util/test/testutil.go` ignored by Semgrep's defaults: zero matches does not establish complete coverage

Counts miss replacements; line numbers move with comments.

08 · YOUR TURN

Add a deterministic tool

Build

1. Pick one defined, repeatable question about your target's source
2. Answer it with a tool instead of the model
3. Save the result so the next run can be compared

Done when

Running it again gives the same answer, and you can see what it covers.

Smallest version

One search rule that lists one kind of call.

Going further

Test cases
Test should-match and should-not-match cases.

Other call shapes
Test ignore behavior and other call shapes, not only the motivating example.

Save the result
Save hashes, versions, raw results, coverage, prior state and limits.

09

Hooks

Run configured behavior at a specific agent lifecycle event.

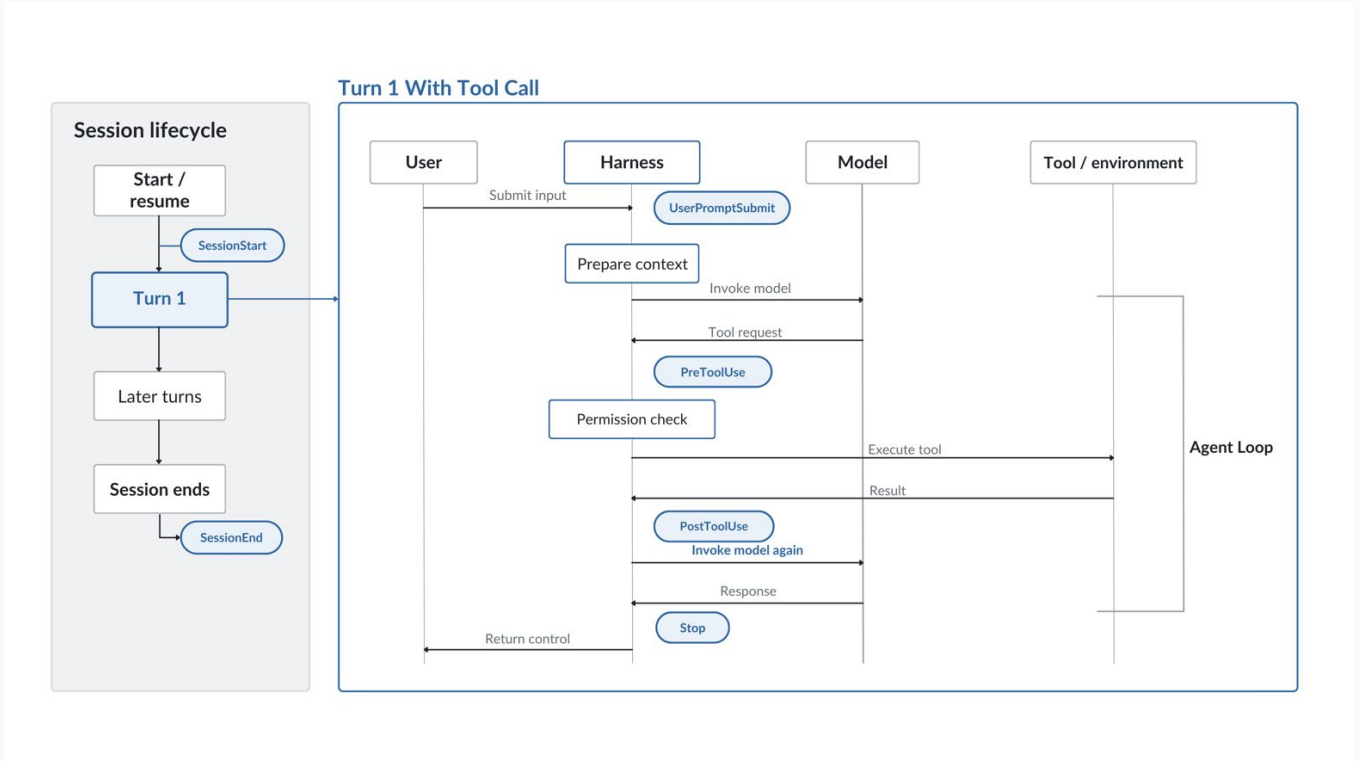
Hooks

- Invoke pre-configured behavior at a lifecycle event.
- Relying on instructions is not sufficient.
- **Observe**: record tool execution metadata.
- **Enrich**: load current research context at session start.
- **Control**: reject a tool request before execution.

lifecycle events

SessionStart	session
UserPromptSubmit	turn
PreToolUse	agent loop
PostToolUse	agent loop
Stop	turn
PreCompact	session
SessionEnd	session

Session lifecycle and agent loop



09 · YOUR TURN

Add a hook

Build

1. Pick one job: observe, enrich or control
2. Choose the lifecycle event it belongs to
3. Configure the handler in your agent

Smallest version

A PreToolUse hook that logs every shell command.

Done when

You can trigger the event and see the result.

ADVANCED

Oracles

An oracle is anything that reveals information about something you don't have access to.

Oracles

- A model does not have domain specific knowledge to accurately answer research questions
- Expose an interface, an oracle, to our harness that has data the model requires to answer our research questions
- An oracle doesn't make security decisions on its own
- It provides data that humans or agents can use to make more informed decisions
- This changed the task from an open-ended question to an agent into a reproducible query

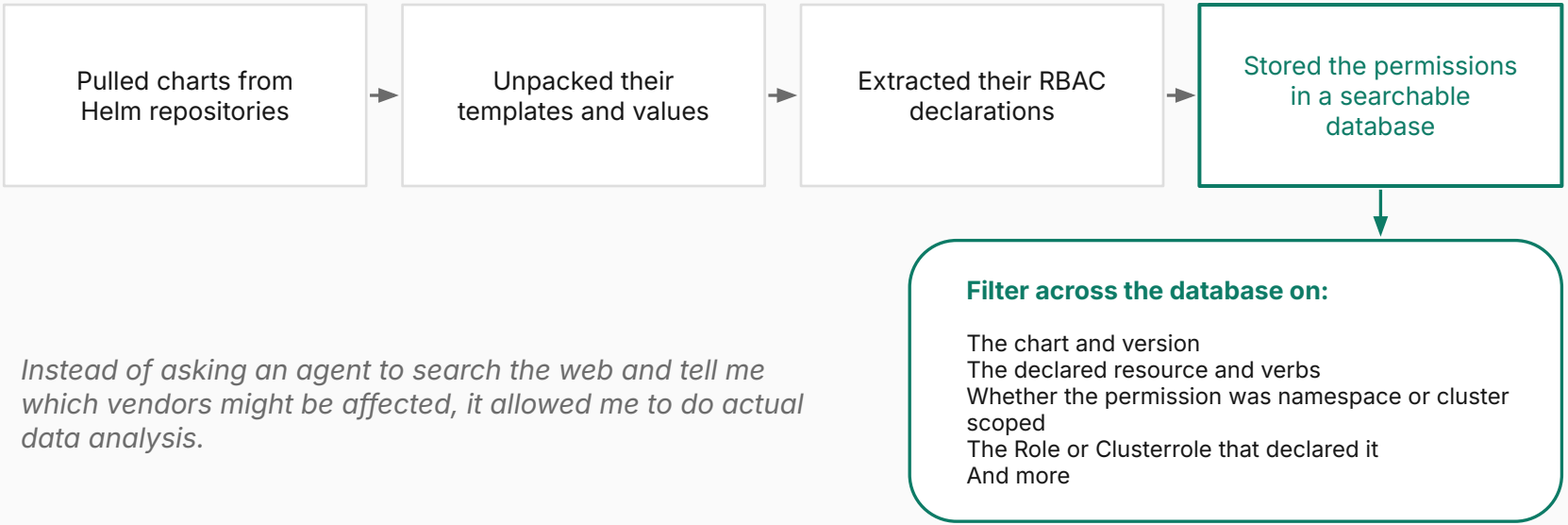
```
rbac-oracle

$ query nodes/proxy
    --verb '*'

chart   chart@1.4.2
verbs   get, create
scope   cluster
role    ClusterRole
```

Case study: Nodes/Proxy RCE

A WEB APPLICATION THAT:



Instead of asking an agent to search the web and tell me which vendors might be affected, it allowed me to do actual data analysis.