

Build Your Red Team AI Crew

One Prompt vs Five Agents

Ariz Soriano / Hebe AU

About us

Ariz Soriano

Associate Director
Red Team Operations

THEOS Cyber Solutions

Why we ran this study

To measure what the pipeline adds
to a well-specified prompt.

Hebe AU

Senior Consultant
Offensive Security

THEOS Cyber Solutions

SECTION 01

Study setup

The question, the conditions and the four tasks

The tool outcomes come first

Strongest demo C3 Chrome: recorder and generator,
9 of 9 automated suites passed.

Weakest outcome C4: incomplete implementations
across all four tools.

The useful question What runs, what is missing,
and what extra work remains?

Four conditions, sixteen scored cells

Condition	Workflow	Model
C1 Claude Code	One comprehensive prompt	Sonnet 4.5
C2 Claude Code	Five roles in one session	Sonnet 4.5
C3 Codex	Five roles via AGENTS.md	gpt-5.6-sol, medium
C4 CrewAI + Ollama	Five sequential role agents	Codestral 22.2B Q4_0

C1–C2 is the closest controlled contrast. C3–C4 also change the platform.

What “five agents” meant in practice

Researcher

Planner

Ops Advisor

Tool Architect

OPSEC Reviewer

C2 followed these roles
inside one Claude session.

C4 used CrewAI agents
that emitted source text.
The harness created files.

The experiment measures the implemented workflow, including its tool access.

The four tool-building tasks

Tool	Implemented scope
Chrome MV3 kit	Capture metadata and generate YAML
GoPhish ICS sidecar	Generate calendar files and record RSVP fixtures
Empire advisor	Score command text using local rules
After-action collector	Join logs and decisions into two reports

SECTION 02

Prompts and agent contribution

What we supplied, and what the stages added

The input was already a specification

Words ¹	PROMPT.md	ACCEPTANCE.md
Chrome	2033	1302
GoPhish	292	190
Empire	254	173
After-action	235	161

¹ Whitespace count including Markdown. Length varies; every task has explicit requirements.

Inside the after-action prompt

- R1** Ingest at least two log formats (e.g. JSON lines + CSV) via adapters
- R3** Correlation engine joins events↔decisions on time window + asset key
- R6** PII/secret redaction pass (regex/rules configurable) before client export

Verbatim excerpts. The prompt also requires both reports, a fixture pack and an OPSEC card.

What C2 added before writing code

Researcher

Assumptions, format references
and unresolved design questions.

Planner

File layout, interfaces, dependencies
and requirement-to-test mappings.

Ops Advisor

Operating constraints and requested
changes to the implementation plan.

12 pre-build documents, 14,250 words across four C2 runs. Document volume is not measured benefit.

A design decision can survive the handoff

After-action	Pipeline contribution	Observed implementation
Time-window join	Choose a ten-minute default	C2 implements configurable 600 s
Redaction preview	Ops requests a dry-run mode	Present, but previews a sample string
Single-prompt comparison	C1 chooses its own five-minute window	C1 and C2 both score 4.6

Added design detail is visible. A quality gain from those particular decisions is unproven.

Tests and scores answer different questions

Tests

Does the implementation run and satisfy the required checks?

Quality score

Correctness, API fidelity, OPSEC, code quality and completeness.

Time

Recorded generation duration.
Full acceptance time is unavailable.

SECTION 03

Tool outcomes

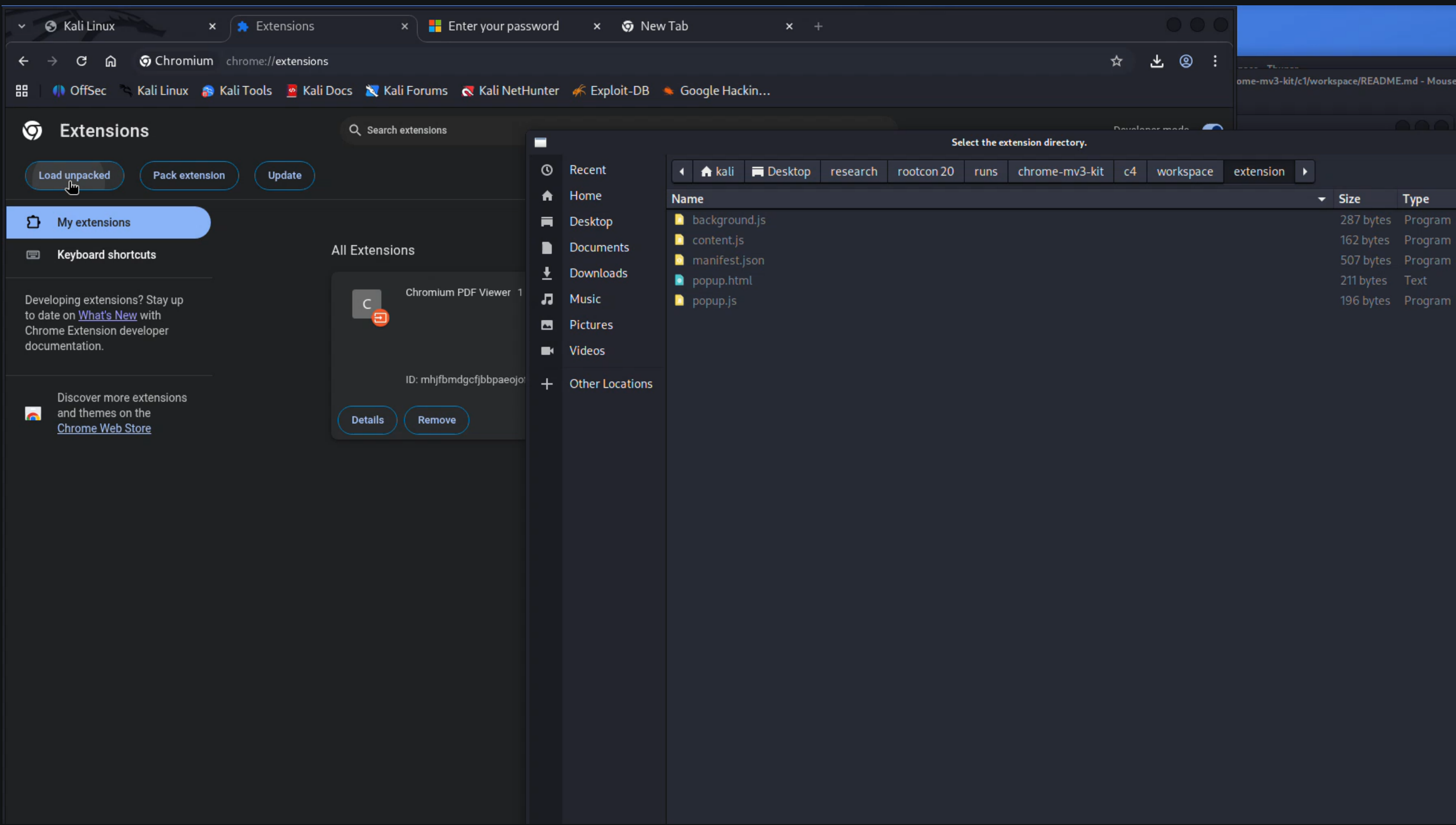
What worked, what was added and what was missing

Demo

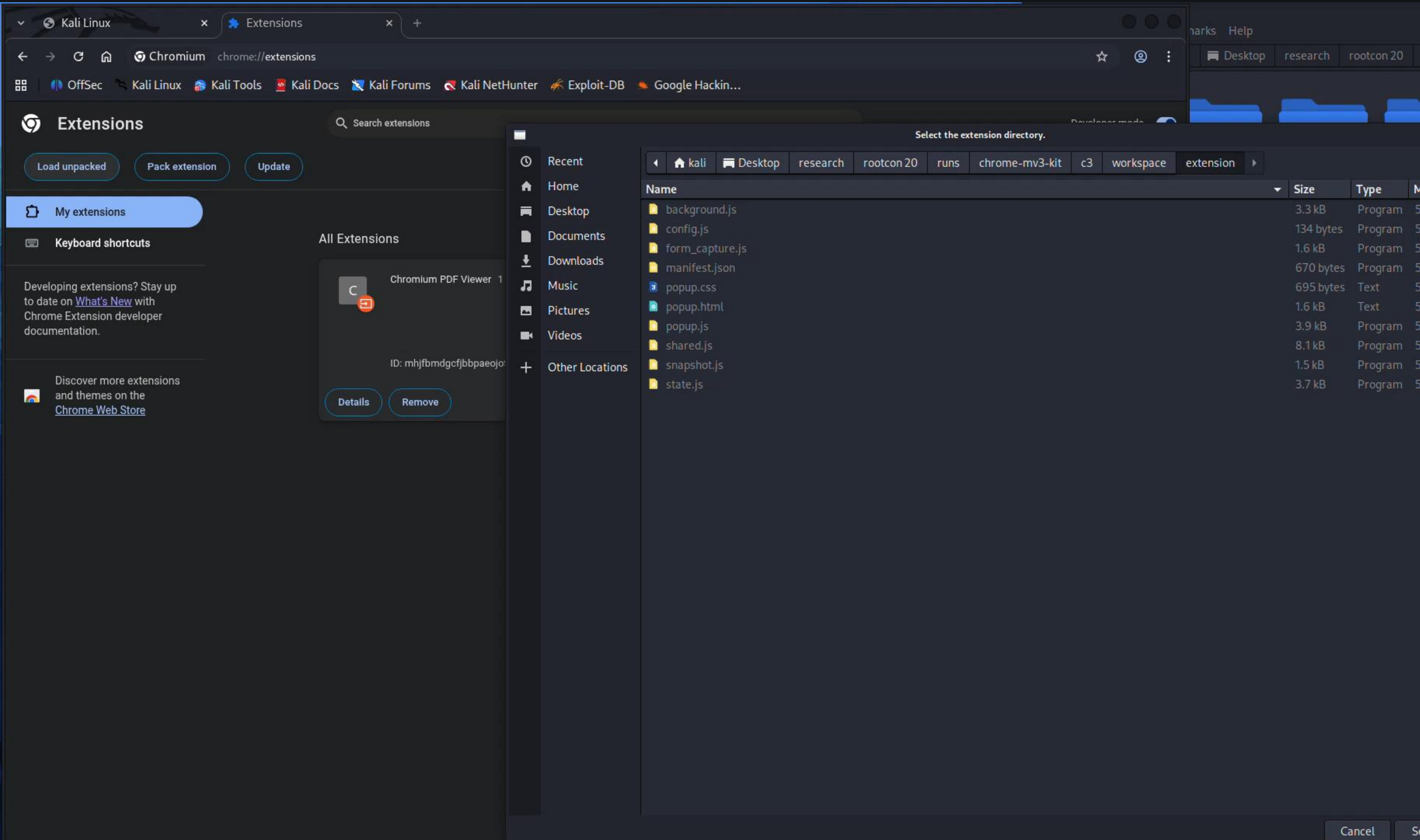
C1 Chrome → C3 Chrome → C4 Chrome

Study artifacts using synthetic fixtures

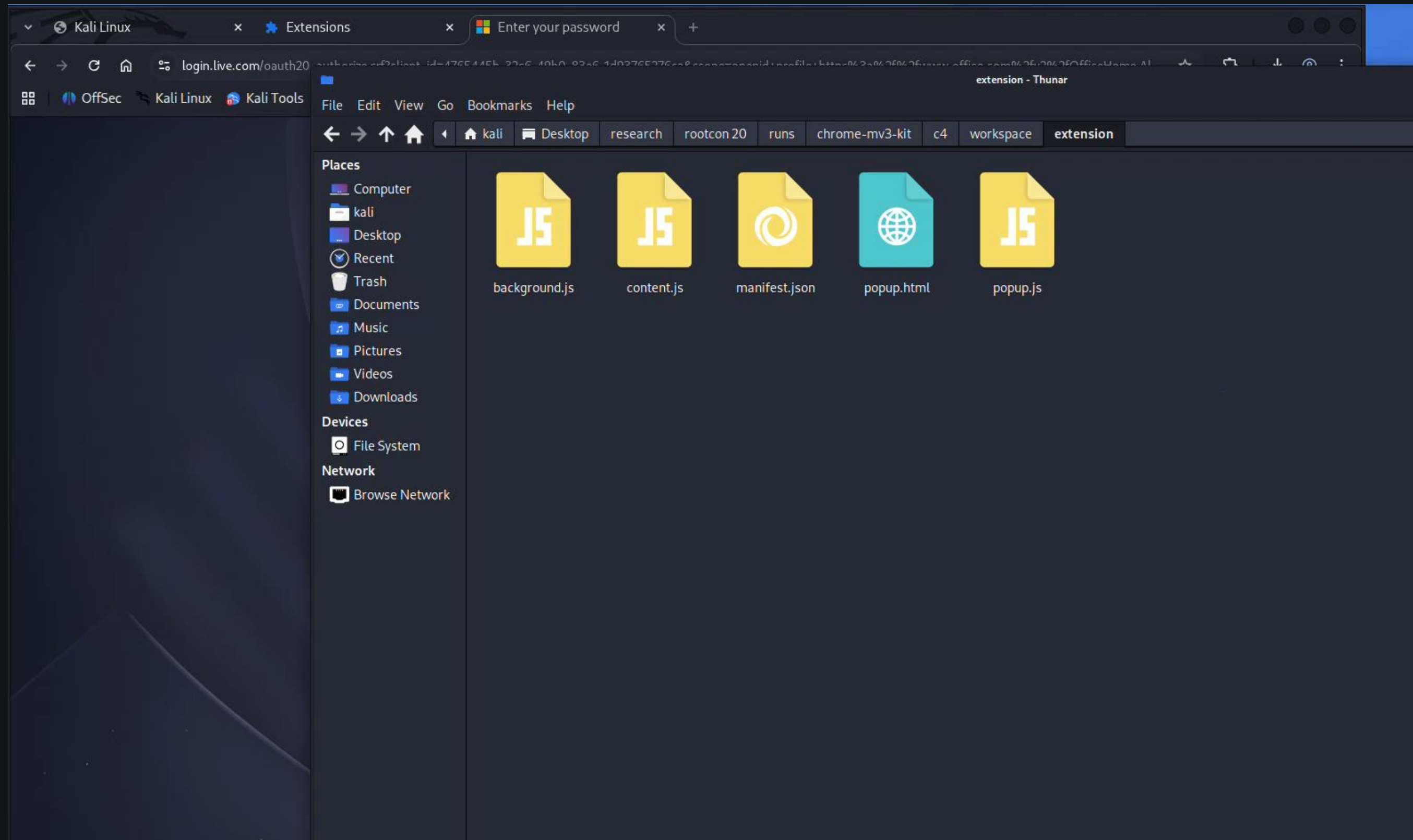
C1 Chrome — Single prompt



C3 Chrome — Codex pipeline



C4 Chrome — Local pipeline



Best Chrome demo: C3

9 / 9

automated suites passed

Recorder lifecycle and export
Form metadata capture
Generator and schema validation
Regression and safety checks

Archived test result. Live multi-origin acceptance A3b remained unverified at scoring.

Chrome: built, extra, missing

Cell	Built	Extra or missing
C1	Recorder + generator	Live-capture schema failure observed ¹
C2	Recorder and popup	Generator syntax error blocks test run
C3	Recorder + generator	Extra scripting and activeTab permissions
C4	Thin files and documentation	Comment-only UI, no generator or tests

¹ Separate prompt-validation observation. All kits target the narrowed lab scope.

C2 built the recorder and broke the generator

SyntaxError:
Illegal break statement

The full npm test command stopped
before the YAML regression could pass.

Recorded failure in phishlet-generator.js. The unpacked recorder remained usable.

C4 Chrome emitted comments

`extension/popup.js`

```
// Handle user interactions in the extension popup
// Implement UI for starting and stopping data capture
// Implement UI for generating phishlets
// Implement UI for saving static HTML snapshots
```

Source excerpt from the scored C4 workspace. No executable popup behavior.

GoPhish: three working sidecars

Cell	Recorded test outcome	Implementation
C1	12 pytest tests passed	Python sidecar
C2	go test ./... passed	Go sidecar
C3	13 pytest tests passed	Python sidecar
C4	No runnable ICS generation	Comment-only generator

The selected C2 tool uses simulated RSVP fixtures. Live response ingestion remains outside scope.

GoPhish: what the sidecar adds

Built

Calendar file generation, RSVP storage, campaign summaries and default-off config.

Added

C2 chose Go. C1 and C3 chose Python.
This is an implementation choice.

Missing

Automatic delivery and live RSVP ingestion were not demonstrated.

Empire: the advisor works offline

12

C1 tests

7

C2 tests

12

C3 tests

Built: command scoring, local rules and explanations.

C4: a PowerShell stub instead of a working advisor.

Live Empire interception and actual EDR response were not tested.

After-action: the clearest complete workflow

Built

JSONL and CSV ingestion, decision joins, client report and internal learning report.

C3 detail

C3 checks missing-field behavior and warns when records are malformed.

Missing in C4

A runnable CLI, required modules and the fixture report workflow.

C1/C2: make test passed. C3: 7 unittest tests passed. C4: tests could not collect.

One fixture, two report audiences

CLIENT REPORT

“Draft coordination contact:
[REDACTED EMAIL].”

Detection Recommendations

“Correlate process creation with
network connections on
high-value servers.”

INTERNAL LEARNING

“One JSONL record was malformed
and required source-side
correction.”

“The database alias was not
represented in the asset
inventory.”

Exact excerpts from the synthetic Northstar fixture. Both reports need human review.

Best and worst, by actual outcome

Finding	Evidence
Preferred Chrome demo	C3: generator suites pass and operator preferred its capture output
Most consistent tasks	GoPhish and after-action: C1–C3 tests pass
Weakest Claude result	C2 Chrome: usable recorder, broken generator
Weakest overall condition	C4: all four implementations incomplete

SECTION 04

Comparison results

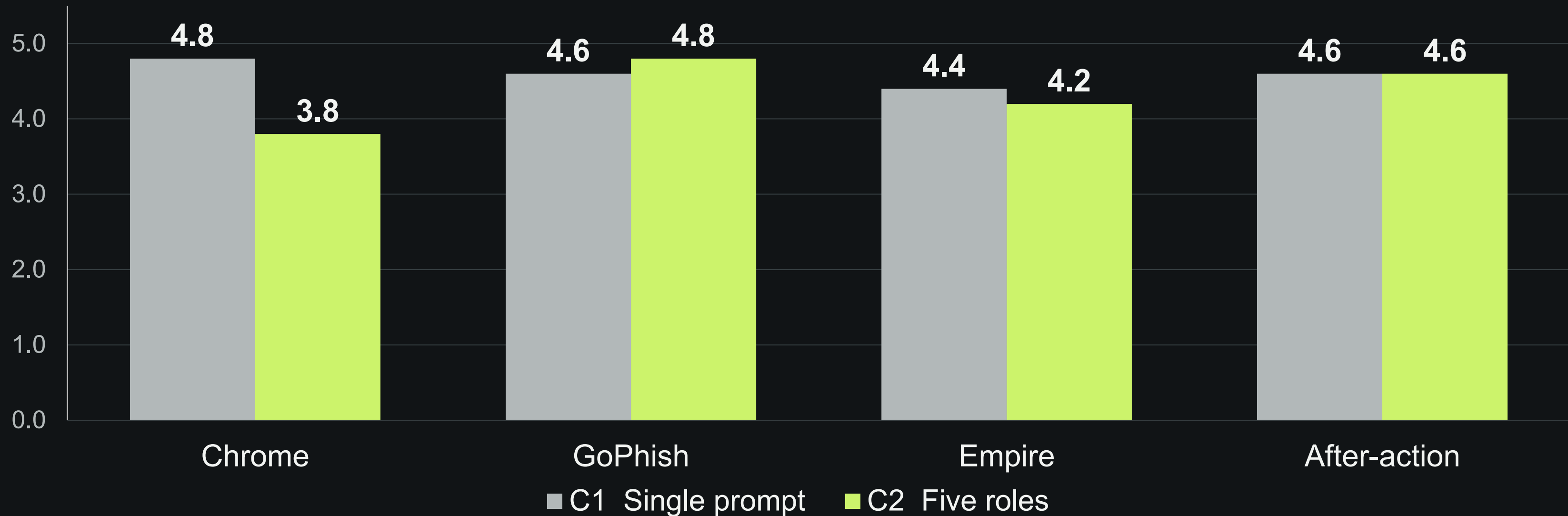
Quality, generation time and the three research questions

Quality scores across the four tasks

Tool	C1	C2	C3	C4
Chrome	4.8	3.8	4.8	1.2
GoPhish	4.6	4.8	4.8	1.2
Empire	4.4	4.2	4.8	1.4
After-action	4.6	4.6	4.8	1.2
Mean / 5	4.60	4.35	4.80	1.25

Five quality dimensions, each 0–5. One unblinded rater. Timing is separate.

RQ1: the Claude pipeline did not improve quality



Mean difference: -0.25 points out of 5. The largest loss was Chrome's generator.

RQ2: the workflow transferred to Codex

+0.45

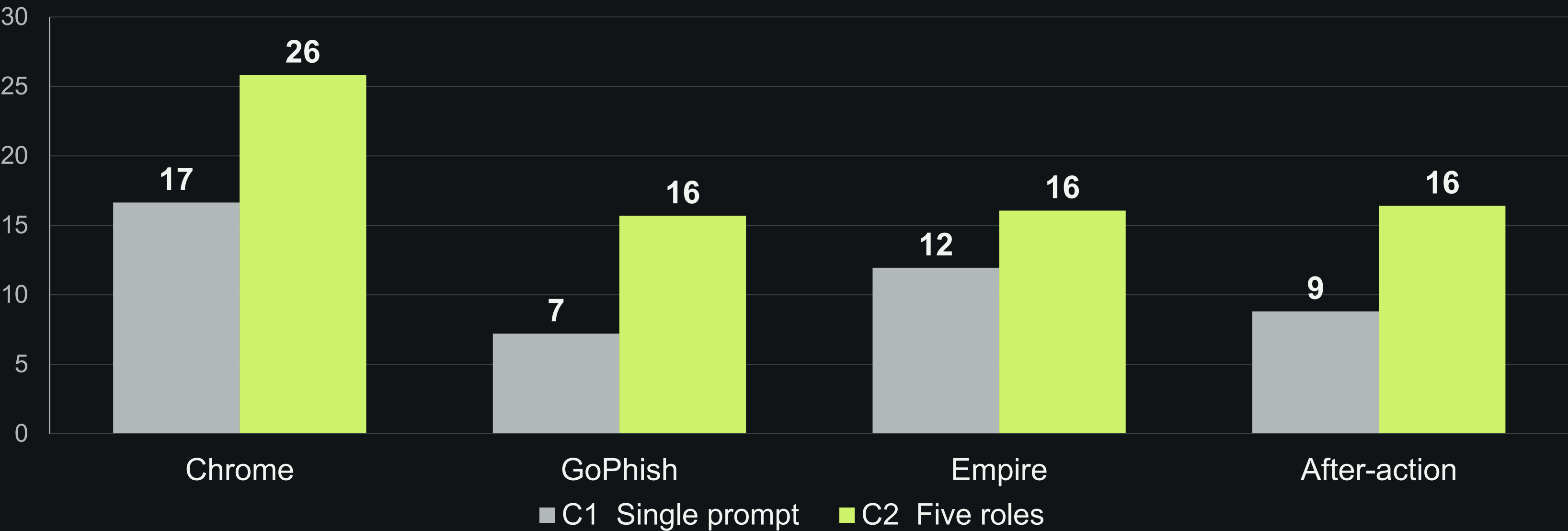
mean quality difference
C3 compared with C2

C3 scored 4.8 on every task.

The model and platform changed.
This is evidence of workflow transfer.

Codex gpt-5.6-sol versus Claude Sonnet 4.5. Four tasks, one selected output per cell.

C2 took longer to generate every tool



Generation minutes. Mean C1: 11.15 min. Mean C2: 18.49 min.

C2 took longer, with a lower mean score

C2 minus C1	Extra minutes	Quality change / 5
Chrome	+9.18	-1.0
GoPhish	+8.49	+0.2
Empire	+4.12	-0.2
After-action	+7.59	0.0

Sum of run durations: +29.38 min (+65.9%). Prompt development and human review time are unmeasured.

RQ3: C4 produced no complete tools

0 / 4

Comment-only source
Missing modules
Tests that could not collect

complete working implementations

C4 lacked the same file-editing and test-feedback loop. No local single-prompt baseline.

SECTION 05

Takeaways and limits

What these results support, and what remains untested

What the evidence can and cannot establish

Study size

Four tasks, selected outputs,
one unblinded scorer.

Shared analysis

Semgrep and ast-grep after generation.
No shared Serena usage.

Operational proof

Fixture checks and offline integrations.
Local inference with private-network logs.

The pipeline may help write the specification

Current evidence

Implementation from an explicit specification and acceptance checklist.

Hebe's hypothesis

Roles may help people turn a rough idea into a useful specification.

Still untested

Whether that reduces missing requirements or saves total human effort.

A follow-up needs both brief and detailed inputs, with the same model and tool access.

The three answers

Claude

The five-role session averaged lower quality and took longer to generate.

Codex

The workflow transferred to these tasks.
Model and platform effects remain mixed.

Codestral

This CrewAI configuration did not produce complete working tools.

The release package

Methodology	Condition configs, personas, score matrix and failure evidence.
Tool candidates	C3 Chrome, C2 GoPhish, C3 Empire and C3 after-action.
Client handoff	Each release folder has an OPSEC card with Detection Recommendations.

Local artifacts prepared. Complete public release and Chrome snapshot replacement remain pending.

Questions

Ariz Soriano / Hebe AU

Build Your Red Team AI Crew

Appendix: runtime and method details

Condition	Recorded runtime
C1 / C2	Claude Code 2.1.259, claude-sonnet-4-5
C3	Codex CLI 0.147.0, gpt-5.6-sol, medium
C4	CrewAI 1.15.18, Ollama Codestral 22.2B Q4_0
Model digest	0898a8b286d5...
Scanners	Semgrep 1.176.0, ast-grep 0.45.3, post-run CLI

C4 network mode in run logs: private-network. Verified egress isolation is not archived.

Appendix: acceptance-count discrepancies

Cell	CSV passes	Itemized PASS rows
Chrome C1	20	21
Chrome C3	20	21
Chrome C4	3	2
GoPhish C4	2	3
Empire C4	3	4

Recount of archived tables, not a rerun. Main slides use specific test outcomes instead.

Appendix: generation duration

Minutes	C1	C2	C3	C4
Chrome	16.64	25.82	18.02	14.77
GoPhish	7.20	15.69	13.73	31.81
Empire	11.94	16.06	13.56	6.93
After-action	8.81	16.40	11.00	2.91

C4 after-action has a stale start time in RUN_LOG. The 2.91 min comes from run metadata.