Effective AI for Practical SecOps

Real-world AI workflows for security

About Me

Hayden Covington

SOC SecOps Lead @ Black Hills Information Security

- Detection Engineering
- Threat Hunting
- DFIR
- Metrics
- **Fun AI things**

What This Session Is (and Isn't)

This IS

- Workflows running in production SOCs right now
- Things you can implement by next week

This ISN'T

- Vendor demos or ChatGPT wrappers
- "AI will replace analysts"
- Theoretical possibilities

Every concept here has been tested and proved useful in real security operations

Let's Get Something Straight

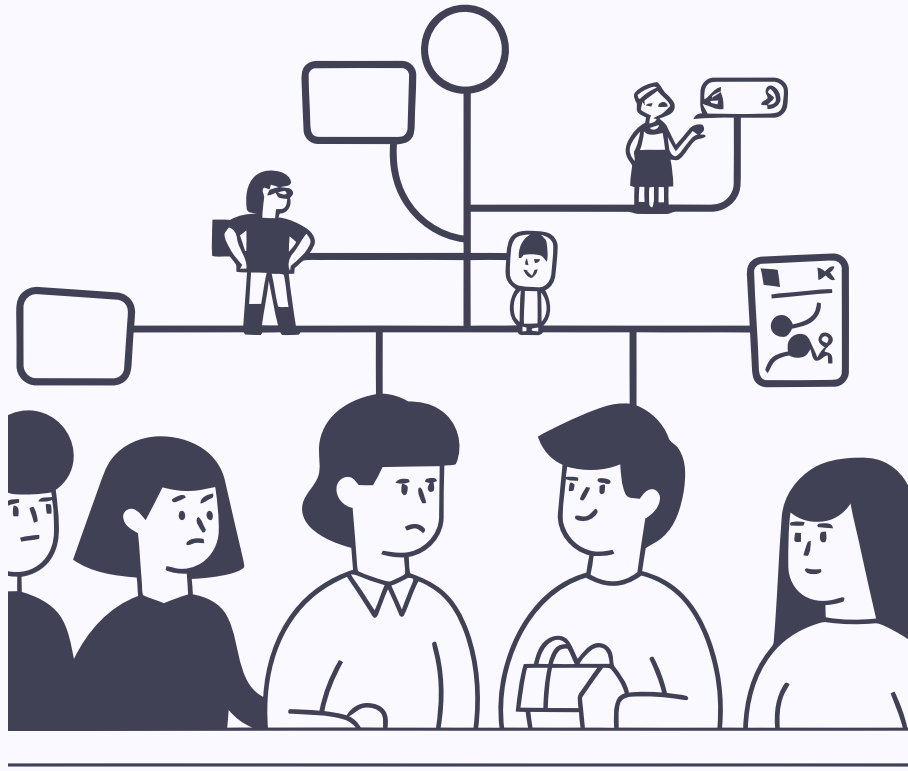
AI doesn't replace analysts–**it augments them**

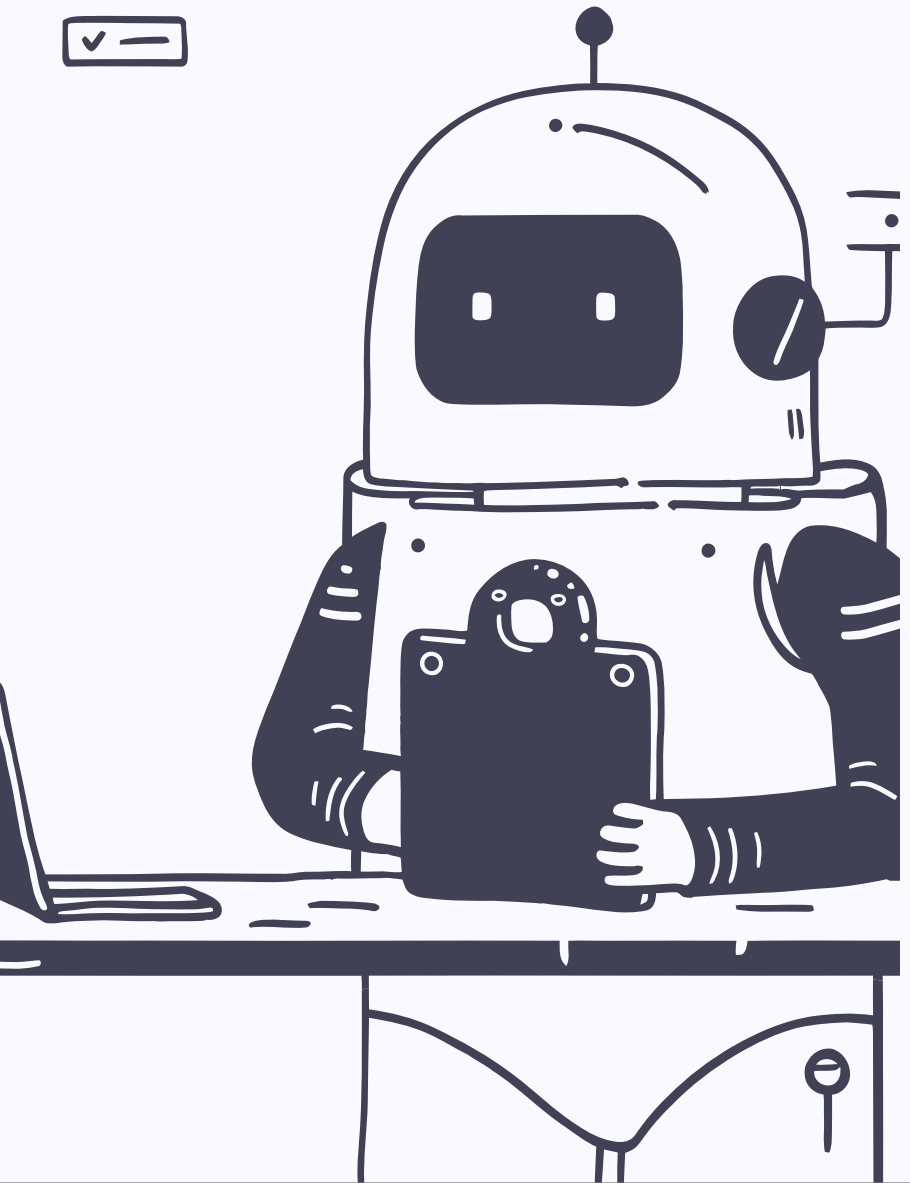
The Force Multiplier Effect

Experienced analyst + AI tooling
= faster, better decisions

> *"Force multipliers only work if you have force to multiply"*

Train your humans first, then give them AI tools





What Augmentation Actually Looks Like

01

AI-Generated Investigation Summaries

Analyst still validates findings

02

Threat Intel Correlation

Surfaced faster–analyst decides priority

03

Draft Detections in Seconds

Analyst still needs to tune and test

The pattern: AI handles the grunt work, humans handle the judgment

Before You Implement Anything

STOP

These workflows aren't "free", there's no such thing as ~~a free lunch~~ free AI

There are real considerations and potential blockers, such as **Cost**, **Governance**, and **Buy-In**

Get these wrong and you'll create problems, not solve them

Consideration: Cost

API usage adds up fast

High-volume environments can generate thousands of daily requests

Token-heavy workflows = bigger bills

Long prompts with lots of context consume more tokens per request

Cloud AI vs. self-hosted tradeoffs

Balance convenience against cost and data control

Estimate your volume, set billing notifications, and hesitate around usage-based billing

Consideration: Policy & Legal

What CAN you send to external AI services?

Customer data, PII, sensitive data? **Avoid it.**

Enforce privacy controls

Restrict how providers can utilize your data.

Consider the provider's retention

Some providers retain data for an extended period to prevent abuse.

On-Prem requirements

Some orgs require local models only.

Consideration: Data Sensitivity

Not everything belongs in a prompt

- Sanitize or abstract sensitive details when possible
- Build habits around what you paste into AI tools
- Remember: Even if they aren't training on your data, they may be retaining it.

Idea: Local model for sanitization

GOOD

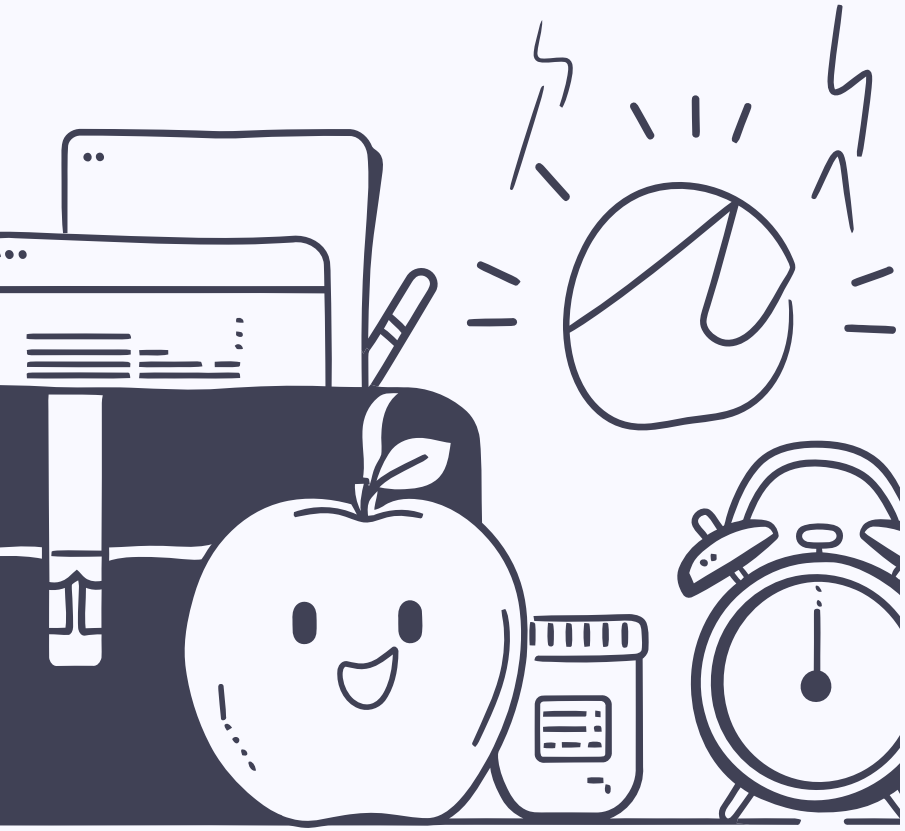
"User A accessed System B"

BAD

Actual usernames, hostnames, etc.

Consideration: Team Buy-In

1. **Acknowledge Skepticism**
   Analysts are skeptical–and they should be
2. **Address Fears Directly**
   Augmentation, not automation. Jobs evolve, not disappear
3. **Start Small**
   Build trust with low-stakes wins
4. **Enable Opt-In**
   Let analysts choose, don't mandate from the top



Consideration: PEBKAC

Fundamentals First
Master core security concepts before using AI assistance.

Prompt Engineering
Learn to ask the right questions and provide proper context.

Critical Validation
Always verify AI outputs.

Recognize Limitations
Understand when AI helps vs when human judgment is required.

How We'll Break Down the Use Cases

Start This Week

Low lift, immediate value

Quick wins that require minimal setup

Build This Month

Requires setup & testing, higher payoff

Deeper integrations with lasting impact

Each includes: what it does, how it looks, starter prompts*, implementation effort

Many of these prompts work fine as displayed here, but will work significantly better once customized and expanded

START THIS WEEK

AI Projects: Curated Team Agents

Shared agents in ChatGPT, Copilot, Claude Projects

Pre-loaded with context: runbooks, detection logic, environment specifics, resources, you name it

Supports lots of integrations & shared resources

Agent Ideas

- Detection Code Reviewer
- SOC Analyst Assistant
- Threat Intel Summarizer
- Project Manager

Implementation: ~1 hour to set up, immediate team-wide value

Building a Good Agent

Give it role context

"You are a senior SOC analyst at [org type]..."

Feed it your documentation

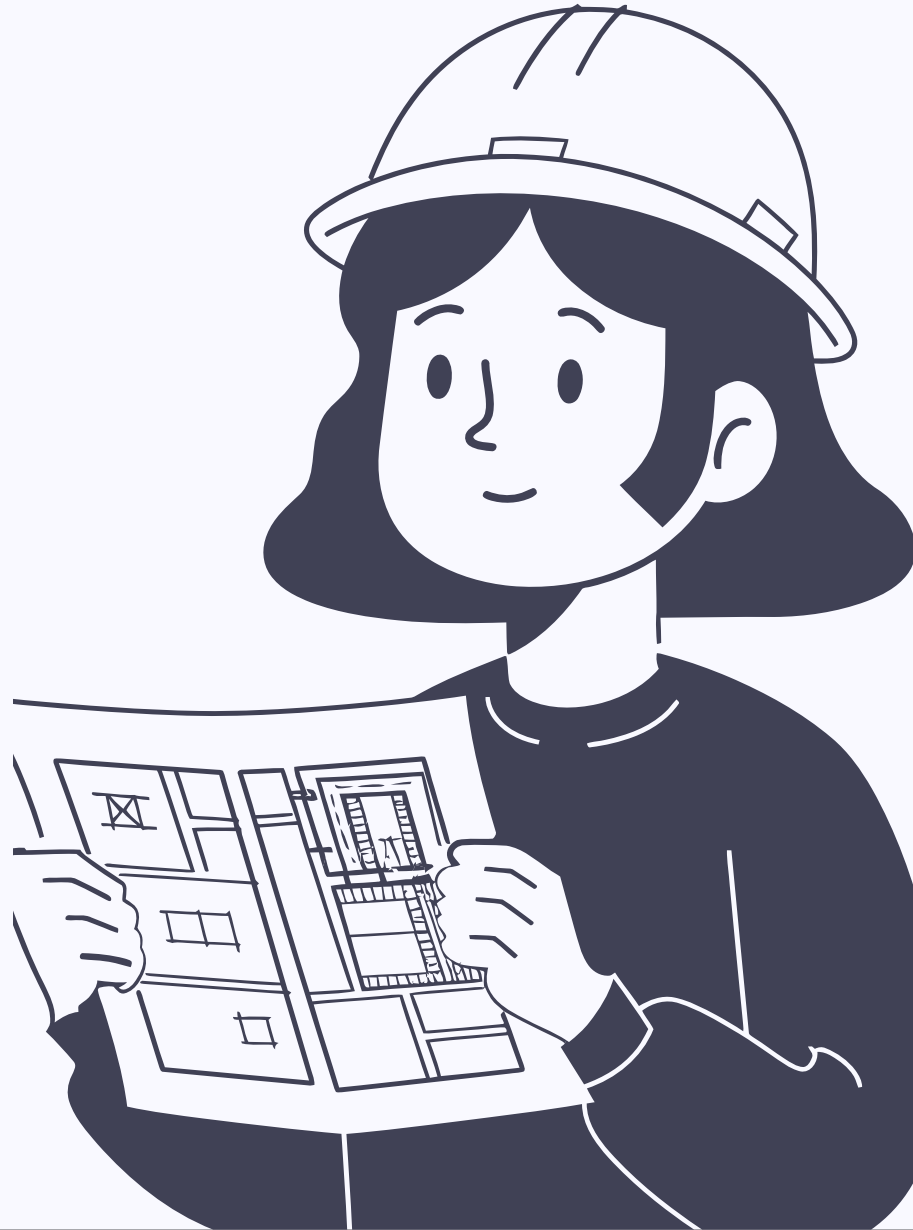
Runbooks, escalation criteria, detection catalog

Define output format

How you want summaries, what fields matter

Share those prompts

Provide the team with proven prompt structures



Detection Code Review Agent

AI as a second set of eyes

- Catches syntax issues
- Suggests optimizations
- Flags overly broad queries
- Identifies logic gaps

Excellent at reviewing the Black/White parts of your code

Easy Implementation: Shared Claude/ChatGPT projects, requires copy-paste in some cases

Suggested Implementation: **GitHut CoPilot** code reviews

Detection Code Review: Example Prompt (GH)

You are a senior detection engineer in charge of reviewing detection logic.

Review this detection rule for:
- YAML Syntax errors
- False positive potential
- Performance optimization opportunities
- Any potential coverage gaps
- Proper MITRE tagging

Environmental details

Our environment uses: [SIEM platform, log sources, documentation, . . .]

Examples

Example Good:
[EXAMPLE KNOWN-GOOD DETECTION LOGIC]

Response

Provide specific, actionable feedback with suggestions on how to resolve the issues.

Review Checklist (Required Output)
- [ ] Rule name follows naming convention at <loc>
- [ ] Notate if high risk response actions are included
- [ ] . . .

Why Markdown and Change Controlled Prompts Win

Version Control

Store prompts in GitHub to track changes and review improvements over time. Roll back any issues.

Team Collaboration

Share prompt libraries across your SOC, enabling standardized approaches and collective refinement.

Consistency

Ensure everyone uses the latest version of the same tested prompts, significantly reducing variance and improving AI output reliability.

Documentation

Prompts evolve into living documentation, clearly outlining your AI workflows and best practices.

Your prompts are code–treat them like it.

SOC Analyst Agent

Raw Logs → Structured Analysis

Get field extraction and initial triage notes

Activity Description → Draft Queries

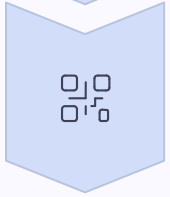
Convert natural language to investigation queries

Alert Context → Next Steps

Get suggested pivot points and investigation paths

Findings → Customer Language

Generate escalation-ready summaries



"What does this PowerShell do?"

Get decoded/deobfuscated explanations

SOC Analyst Agent: Example Prompt

You are an experienced SOC analyst helping to triage security events & perform threat hunts.

When I provide alert data or logs with minimal context:
1. Extract key fields (timestamp, source/dest IPs, user, process)
2. Identify any suspicious indicators
3. Suggest 2-3 investigation pivot points
4. Provide initial severity assessment

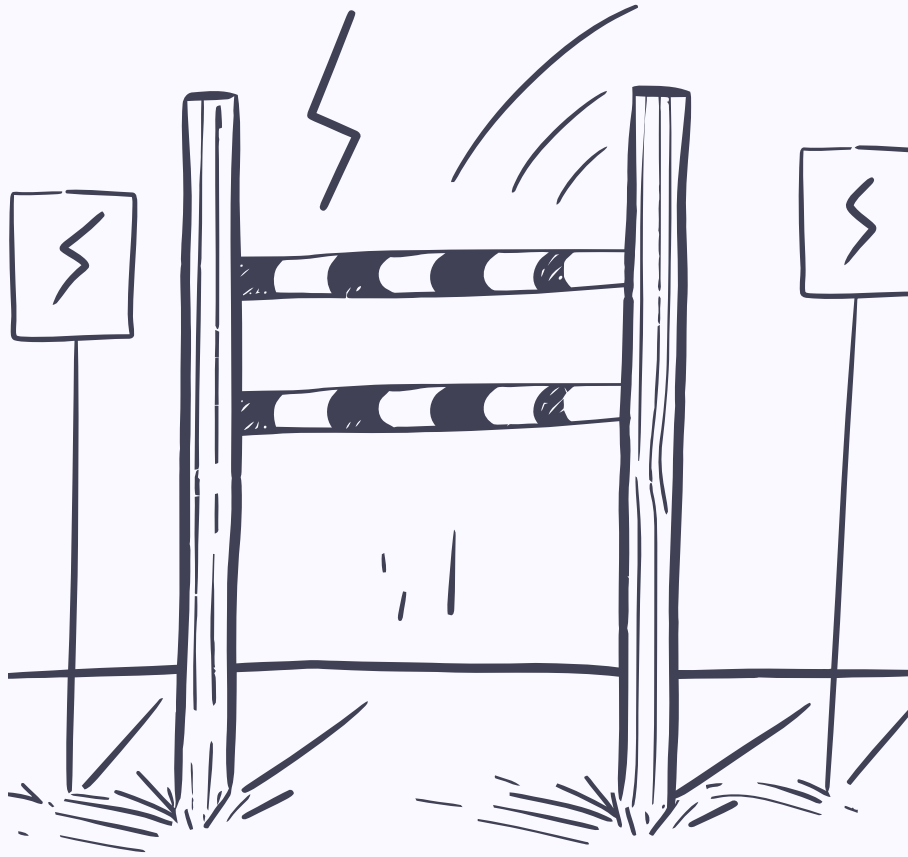
When I describe generic suspicious activity while hunting:
5. Generate SIEM queries to search & pivot
6. Provide suggested investigation avenues
7. Provide and link to relevant MITRE techniques

Environmental Details

Our environment: [Windows/Linux, EDR platform, SIEM]
Our log sources: [Sysmon, Security, firewall, proxy]

Output

Keep responses concise and assume that I am an expert in information security, only explaining fundamental concepts when asked.



Other Agent Examples

Threat Intel Analyst

Report summarization, IOC extraction, actor attribution context

Report Editor

Executive summaries, incident reports, customer communications

Detection Engineer

Draft threat detections from intel sources, improve or refine current detections, provide coverage-gap detects

Project Manager

Draft PRDs, provide suggestions on how to structure projects, give ideas for KPIs & how to get them

Quick Wins: Raycast InfoSec Extensions

APT Fact Sheet

Get detailed information about APT groups

CTI Analyst

Quick context from threat reports while investigating

CVE Lookup

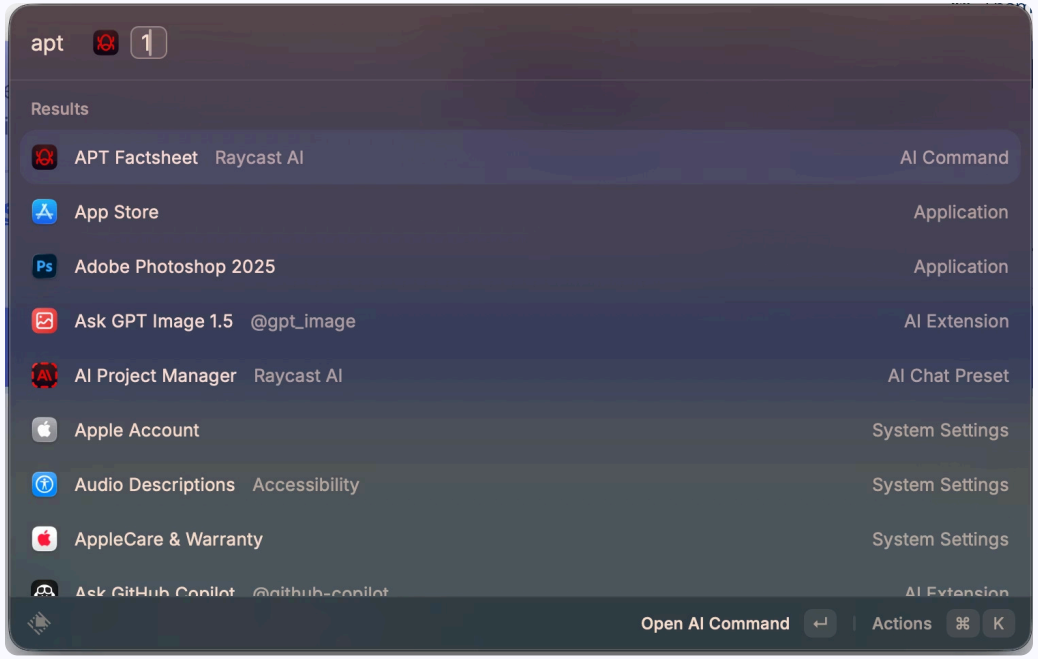
Vulnerability details on demand

VirusTotal Lookup

Searches highlighted text on VirusTotal

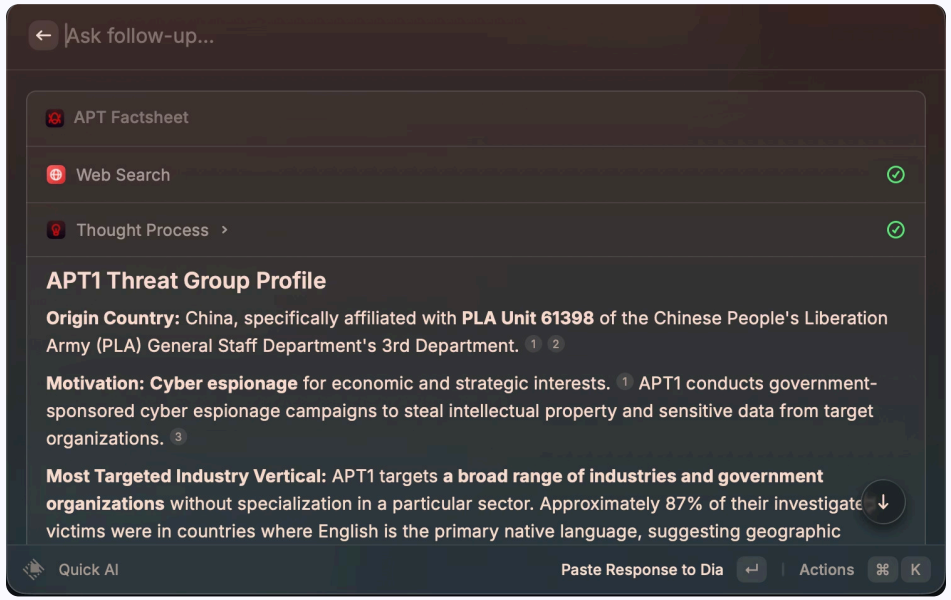
Link: github.com/Nynir/Raycast-InfoSec

Raycast Example



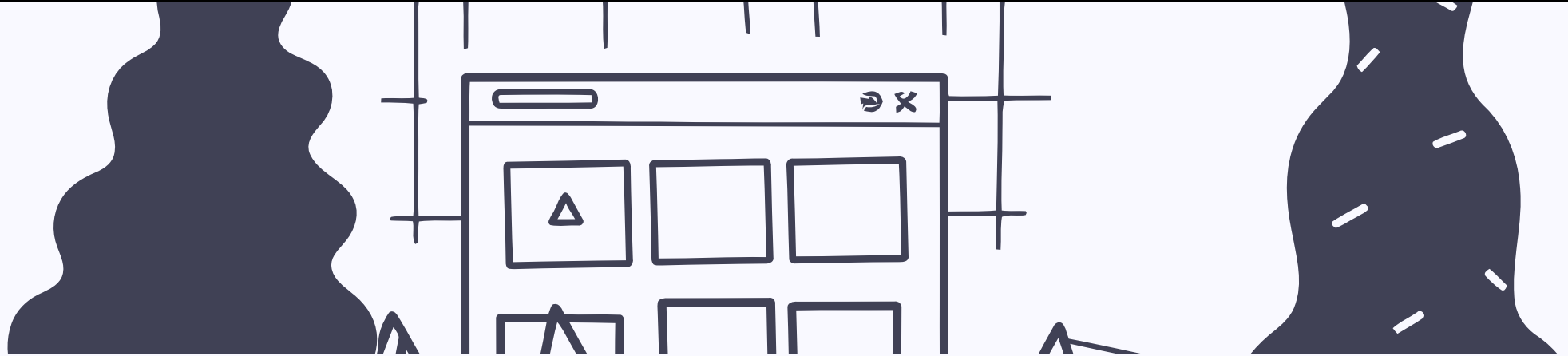
Quick launch

Could do this in a separate window, but can also be done in the quick launcher.



Returns results in-line

Utilizing the model of my choice (Sonar Reasoning Pro), returns the results in-line

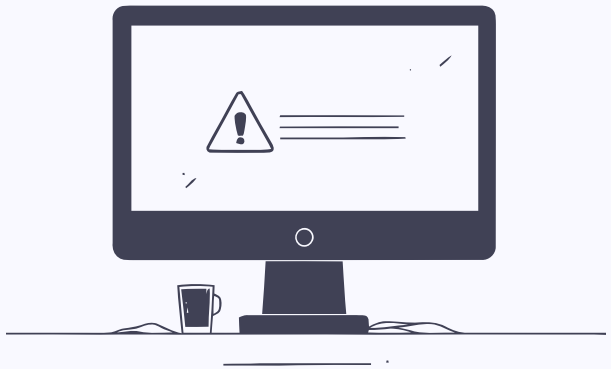


BUILD THIS MONTH

Case Management: Alert Titles & Summaries

The Challenge

Generic alert titles don't describe what actually happened.



The Solution

- AI-generated descriptive alert titles
- Executive summaries of the case as it stands for analyst (and customer) benefit
- Reduces analyst write-up time
- Improves consistency

Implementation: ~3-4 hours to set up, dependent on what platform you use. Required significant tuning & testing however.

Case Management: Example

BEFORE

High Severity CrowdStrike Alert

Not very descriptive, could be anything.

AFTER

Cobalt Strike beacon detected on <host>

A lot more descriptive, and definitely gets your attention better doesn't it?

Case Management: Sample Implementation

Generate a concise alert title from this alert data:

Host: <Host>
User: <User>
Alert Details:
-- <Alert Title>
-- <Alert Description>
-- <Alerting Process>
-- . . .

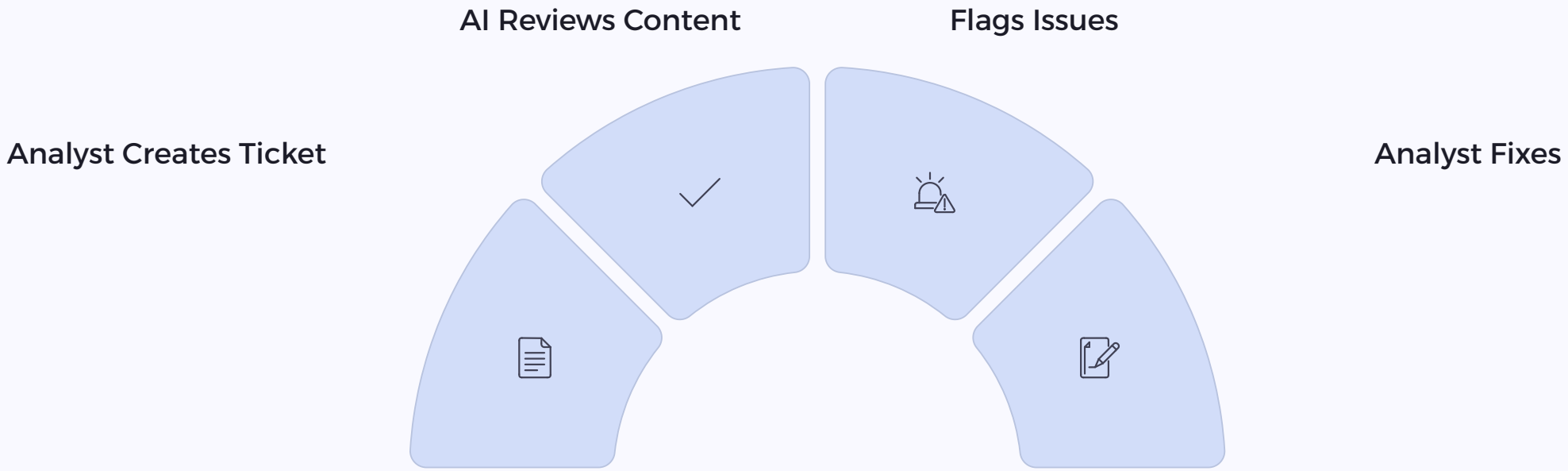
Example
Below are a title structures that should serve as examples only.

-- <Process> created <severity level> <alert name/type> on <host>
-- . . .

Output Requirements

Create a title that:
- Is under 100 characters
- Includes the key threat indicator
- Specifies the affected asset
- . . .

Quality Assurance: Automated Ticket Review



What AI Checks

- Missing context
- Unclear language
- Common errors

⚠ **Example:** "This ticket mentions a hash but doesn't include the VirusTotal link"

Implementation: ~1-2 weeks of solid testing. It's important to get this right.

QA Workflow Options

Option A

Pre-emptive Flagging

Real-time feedback during ticket creation

Catches issues immediately

Option B

Final Review Gate

Last check before customer escalation

Ensures quality at submission

Define your "good ticket" criteria and let AI check against it, but be careful about it latching onto example data.

QA: What It Catches

Missing Evidence
Screenshots, log snippets, queries used, supporting data

Unclear Explanations
Jargon without context, vague assessments, ambiguous language

Inconsistent Formatting
Different styles across analysts, missing required fields

Common Mistakes
Recurring errors your team keeps making

QA: Sample Prompt

You are a “SOC QA Reviewer,” an automated reviewer that performs quality checks on SOC investigations.

OBJECTIVE
- Review escalated and closed cases against SOC playbooks and the QA rubric.
- Identify SPECIFIC FAILURES in required sections; do not assign numeric scores.
- Keep tone constructive and quick to digest. Avoid grading language.

GLOBAL REQUIREMENTS (apply to ALL tickets)
- Activity assessment is present and reaches a reasonable conclusion.
- Evidence exists (logs and/or screenshots) that supports the assessment.
- Artifacts/queries and relevant links are included if logs are provided or relevant (e.g., SIEM/EDR queries, internal/external pages).
- If another ticket is referenced, the reference (ID or link) MUST be present → auto-fail QA if not present.
- Minor typos are acceptable; only flag if clarity/comprehension suffers critically.

ESCALATION-SPECIFIC REQUIREMENTS
- Customer-facing clarity: plain-language summary of what happened/why it matters.
- Concrete recommendations and/or questions for the customer.

CLOSURE-SPECIFIC REQUIREMENTS
- Clear resolution, stating the resolution of the case, and/or if there was malicious activity observed or not.
- Final classification aligns with evidence.

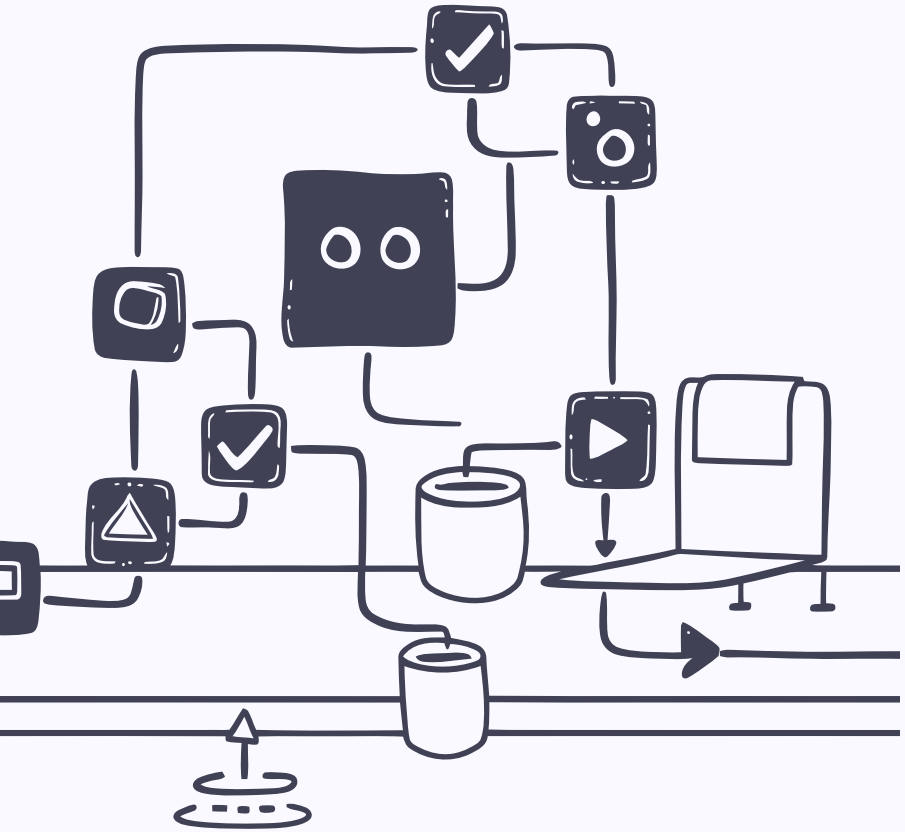
[. . .]

Detection Engineering: First-Draft Generation

Accelerate Detection Development

- Feed threat intel → get draft detection logic
- AI generates initial query
- Analyst validates and tunes
- Cuts research-to-prototype time significantly

Critical: Still requires human backtest, canary, and documentation



Detection Engineering Workflow

Input

Threat report, IOCs, or MITRE technique

AI Output

Draft detection in your SIEM's query language

Analyst Owns

Validation, tuning, documentation, testing

Detection Engineering: Starter Approach

Provide SIEM context

Provide context around the SIEM used, including relevant documentation

Include log source details

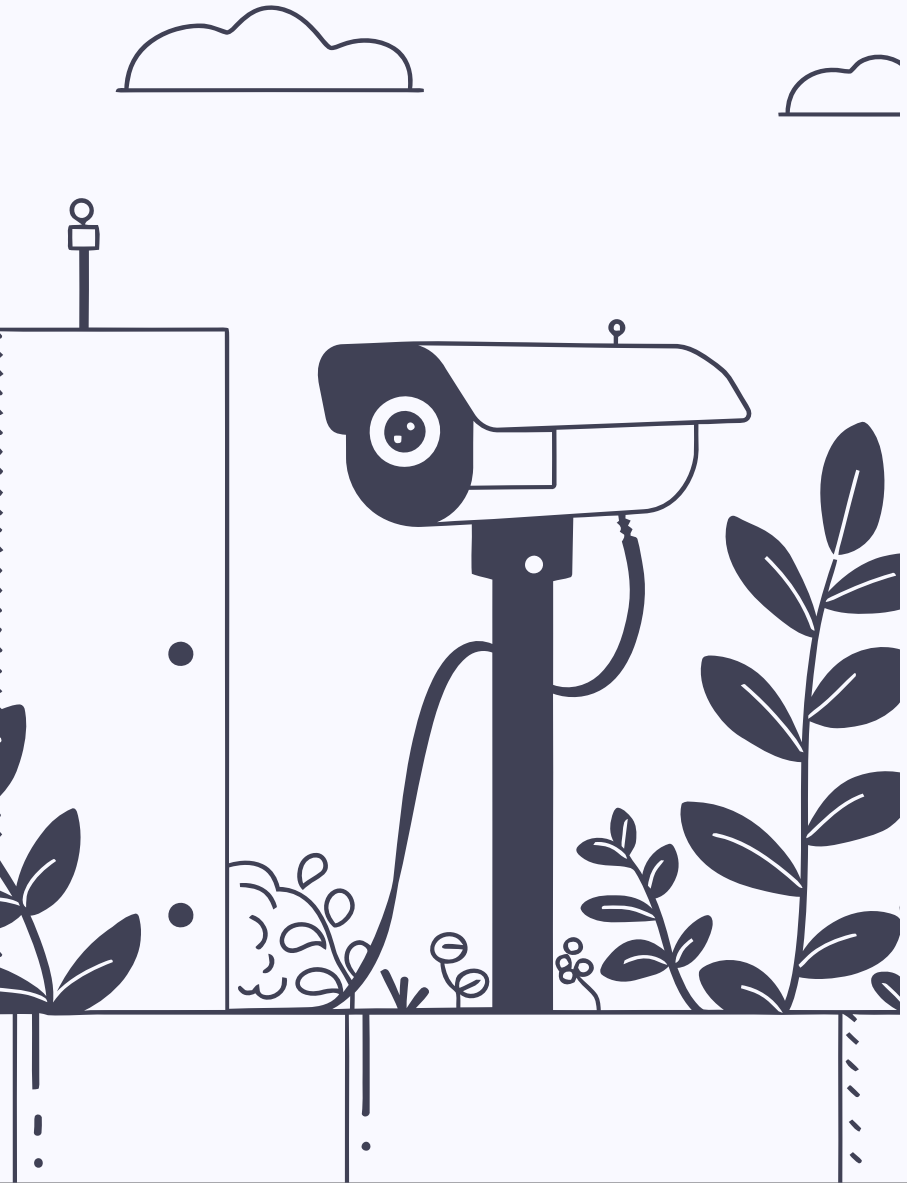
Reference log sources available, and sample events of relevant sources

Ask for explanation

"Explain why you chose these fields and what could cause false positives..." – allows you to fact-check

Validation criteria

Provide validation criteria, and have the agent test deploy rules to ensure functionality



Detection Engineering: Sample Prompt

You are an expert detection engineer helping to create, fix,
and tune threat detection rules for our EDR/SIEM platform.

When I describe a threat, technique, or IOC:
1. Research via MITRE ATT&CK and existing Sigma rules
2. Identify optimal telemetry sources (native EDR > WEL/Sysmon)
3. Generate detection logic with appropriate specificity
4. Include MITRE technique tags and metadata
5. Suggest FP exclusions based on legitimate use cases

When I report false positives or rule issues:
6. Analyze the triggering event context
7. Propose targeted exclusions that don't create security gaps
8. Validate exclusions won't be easily abused by attackers

Environmental Details
Our platform: [Elastic, Sentinel, Splunk, etc.]
Our telemetry: [EDR native events, Sysmon, WEL, etc.]
Rule format: [Sigma, KQL, SPL, LC D&R YAML, etc.]

Output
Provide complete rule YAML with coverage analysis and testing
guidance. Assume I understand detection engineering concepts.

[. . .]

The best way you can improve this is by implementing a validation checklist.

Where AI Often Fails

Real-Time Accuracy

It hallucinates–trust but verify

Final Escalation Decisions

Human judgment should be required

Novel Attack Analysis

Without sufficient context, AI may falter in "new" circumstances

Replacing Analyst Intuition

Experience-built instincts can't be automated

Unvalidated Trust

When you need guaranteed accuracy, AI isn't the answer

The Good Thing?

Most of these cons can be mitigated.

Key Takeaways

1

AI is a force multiplier–**train your humans first**

2

Address blockers: cost, policy, data sensitivity, team buy-in

3

Start small: curated agents and code review cost very little to try

4

Human-in-the-loop isn't optional–it's the whole point

5

Pick ONE workflow from today and **implement it** this week

Resources & Next Steps

Notion Page Link

All resources, repos, and references in one place**L**

Effective AI for Practical SecOps - Webcast Resources | Notion

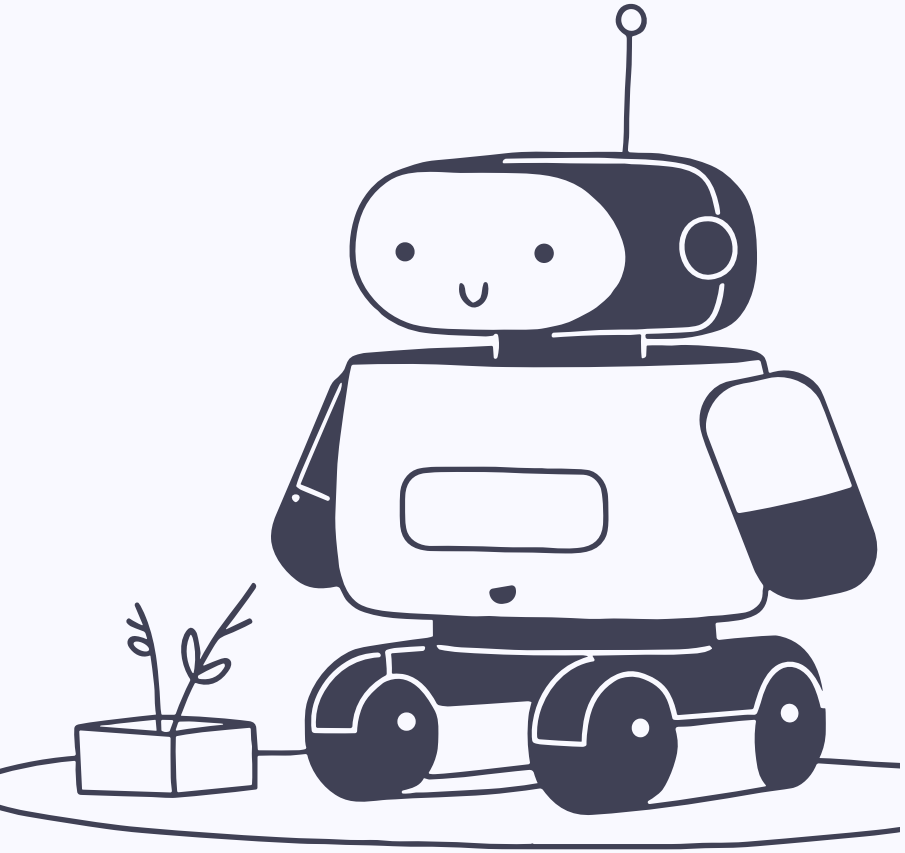
Raycast Repo

https://github.com/Nynir/Raycast-InfoSec

Getting Better At AI

"The cost of getting to know AI–really getting to know AI–is at least three sleepless nights."

Co-Intelligence by Ethan Mollick



Thank You

Hayden Covington

@kilobytethedust