

RUNBOOK

AI-Assisted Incident Response

First-response patterns: log triage, runbook augmentation and hypothesis generation - plus the points where a human must decide.

The first fifteen minutes

When an alert fires, the scarce resource is not intelligence - it is organised attention. AI earns its place in incident response by compressing the search space: it reads the noisy logs, correlates the timeline and drafts candidate explanations while the responder keeps command of what is actually done.

01 Contain first, analyse second

Follow your standard containment runbook before any AI-assisted analysis - isolate, preserve, notify. Model time is cheap; the window where containment still works is not.

02 Triage logs through the model

Feed the alert window's logs - on self-hosted inference if they contain sensitive context - and ask for a structured summary: what changed, what is anomalous, what is likely noise. Verify every claim against the raw lines it cites.

03 Generate hypotheses, ranked

Ask for candidate root causes ordered by evidence in your data, not by generic likelihood. The model's job is to enumerate; your job is to falsify.

04 Augment the runbook

If the incident matches a known pattern, have the model pull the relevant runbook steps and fill in observed values. The responder executes - the model does not touch production.

05 Draft the timeline as you go

Let the model maintain the incident timeline from your verified actions. Post-incident review then starts from a written record, not from memory.

Where the human decides - always

- ☐ Declaring an incident - severity and scope are accountability decisions, not text completion
- ☐ Any action on production systems - restarts, isolation, rollbacks run through human hands with change records
- ☐ Customer or authority communication - wording under legal and contractual weight is written by people
- ☐ Eradication and recovery steps - a model suggests order, a responder confirms each precondition before executing

We run our security operations on Wazuh with this split: detection and first drafts are automated, every consequential action is a human decision with a log.

After the incident

The model's timeline and your verified actions become the post-incident review input. Ask the model to draft the report structure - facts, impact, contributing factors, preventive measures - then rewrite it until it is true. The pattern that pays off over time: feed confirmed findings back into detection rules and runbooks, so the next response starts further ahead than this one did.

This document is part of the UHL resources collection. Direct download and related work:
<https://uhl.cloud/en/resources/>

