From Raw Logs to Actionable Intelligence

An agentic pipeline that reads raw security logs and writes the incident report for the SOC team.

Hadi Al Shmaissani · First Agentic AI & Cybersecurity Hackathon in Lebanon (Technologia × 42 Beirut) · 2025

THE PROBLEM

A SOC team gets more log data every day than anyone can actually read. The events that matter, like a suspicious IP or a host that keeps reappearing, sit hidden among thousands of routine ones. Tools that sort this out automatically (SIEM and SOAR platforms) do exist, but they cost a lot and take real effort to run, so smaller teams often go without. That leaves analysts doing the filtering by hand.

WHAT I SET OUT TO BUILD

The goal was simple to state: put in raw logs, get back a short report saying what is worth looking at and why. No dashboard to babysit, no manual lookups. I also wanted it assembled from cheap, accessible parts rather than an enterprise platform, partly to prove the approach does not need an expensive stack to work.

HOW IT WORKS

The whole thing runs in n8n, a low-code automation tool, with a language model handling the final write-up. Logs arrive through a webhook and pass through a few stages:

Log ingestion (webhook) → Parse & extract indicators → Deduplicate → Threat-intel enrichment → Filter malicious → LLM report → Deliver to SOC

- The logs are parsed for the indicators I care about, mostly source IPs, and duplicates are dropped so the same address is not looked up twice.
- Each remaining IP is checked against threat-intelligence reputation data to see whether it is already known to be malicious.
- Anything that comes back clean is discarded, which leaves only the indicators worth a person's time.
- The model then takes those findings and writes a plain incident report: what was found, and what to do about it. That report gets emailed to the team.

Built with: n8n · webhooks · threat-intelligence reputation lookups · an LLM (Gemini) for the report · automated email delivery

THE PART THAT WAS ACTUALLY HARD

The plumbing was not the difficult bit. Getting the model to behave was. A security report is useless if it changes shape every run, so I pinned the output to a fixed format and kept tuning the prompt until it stopped improvising and produced the same structure every time. Wiring the stages together in n8n, the IP extraction, the dedup, the enrichment, all of that was routine once the report format was nailed down.

HOW IT TURNED OUT

By the time the hackathon closed it ran end to end: drop in a log file, get a report back. The thing that stuck with me was how far a low-code workflow plus a tightly-constrained model gets you. It covered most of what a basic SOAR setup does, without the price tag or the setup. The real lesson was about trust: a model is only worth using here if its output is reliable enough to act on, and that is where most of my time went.

WHY I AM INCLUDING THIS

The shape of the problem shows up far beyond security. Take in messy operational data, clean it automatically, and hand someone a short summary they can act on. A good amount of a ministry's monitoring and reporting work looks like that too, which is why I thought this was a relevant thing to share.

Prepared as an optional writing sample for the MITAI internship application. Built as a team entry at the first Agentic AI & Cybersecurity Hackathon in Lebanon (Technologia × 42 Beirut), 2025.