Title: AI-Augmented Incident Response Playbook System — Mini Agent
Author: Mary Preethi R
Date: 13 November 2025

1. Introduction

Traditional incident response playbooks are static and often struggle to adapt rapidly to new attacker techniques. This project demonstrates an AI-augmented approach to incident response: a Python mini-agent that reads alert data, analyzes it using deterministic heuristics (mock LLM), recommends prioritized, phase-mapped response steps, and allows an analyst to save accepted recommendations for auditing. The prototype is designed to be safely extended to a real LLM integration.

---

2. Objective

Build a reproducible Python prototype that:

- ingests simulated alerts,
- classifies and scores threat severity,
- maps recommendations to IR phases (Containment, Eradication, Recovery, Post-Incident),
- allows analyst interaction and saves accepted actions, and
- is ready to accept an LLM (OpenAI) integration later.

---

3. System Design & Implementation

3.1 Architecture (short)

The system is a single interactive Python script (agent.py) that:

1. Loads sample alerts from alerts.json.
2. Computes heuristic analysis (threat category, numeric risk score, severity label, confidence).
3. Produces phase-mapped recommendations via a rules engine (mock LLM).
4. Displays recommendations to the analyst in an interactive CLI.
5. If accepted, saves the analysis to agent_results.json for audit.

3.2 Key components

- **agent.py** — interactive CLI with deeper IR logic (threat categorization, scoring, phase mapping, save).
- **alerts.json** — sample alert dataset used for demonstration.
- **agent_results.json** — accumulated saved analyses used for evidence and reporting.
- **README.md** — setup and run instructions.

3.3 Implementation highlights

- **Threat categorization**: rules map alerts to categories such as *Credential Compromise*, *Malware*, *Data Exfiltration*, *Reconnaissance*, or *Other* based on alert type and description keywords.

- **Heuristic risk scoring (0–100)**: combines indicators including alert type, description keywords, known-bad IPs, sensitive hosts, and file indicators. Score→severity mapping: ≥75 Critical, ≥50 High, ≥25 Medium, otherwise Low.
- **Confidence score**: a simple fraction of matched indicators (rounded), clamped to avoid extremes.
- **IR phase mapping**: recommended actions grouped by Containment, Eradication, Recovery, and Post-Incident so the analyst sees lifecycle steps, not just ad-hoc actions.

---

4. Example run and outputs

Example 1 — Suspicious login

Input (summary): Multiple failed logins followed by a successful admin login from an external IP to web-server-1.
System output:

- **Threat category:** Credential Compromise
- **Risk score:** 75 → **Critical**
- **Top containment actions:** Isolate host; Force password reset; Block source IP
- **Saved:** Yes (example saved object in agent_results.json)

Example 2 — Malware detected

Input (summary): Endpoint AV flagged invoice.pdf.exe on PC-23.
System output:

- **Threat category:** Malware
- **Risk score:** 65 → **High**
- **Top actions:** Quarantine endpoint; Run full AV/EDR scan; Identify lateral movement

(Include two screenshots here: one showing the interactive CLI with the saved analysis, one showing agent_results.json opened in editor showing a saved JSON object.)

Suggested screenshot captions:

- *Figure 1 &2— Interactive agent output after saving analysis for a suspicious login.*

```
=== AI-Augmented Incident Response — Interactive (Deeper IR Logic) ===
Choose an option:
  1) List sample alerts
  2) Analyze a sample alert by ID
  3) Create and analyze a custom alert
  4) Show last results file path
  5) Exit
Enter choice (1-5): 3

Create a custom alert (press enter to skip a field):
Alert type (e.g., Suspicious login, Malware detected): suspicious login
Short description: I logged into my system this morning and found a new file , which wasnt created b
y me .
Host (e.g., PC-23, web-server-1): pc-20
Username (if applicable): Preethi
Source IP (if known): 10.10.1.7
Destination IP (if known): 172.16.10.50
File name (if applicable): out_irl

--- ALERT ---
id: 4
timestamp: 2025-11-14T19:17:16.336410+05:30
alert_type: suspicious login
description: I logged into my system this morning and found a new file , which wasnt created by me .
host: pc-20
username: Preethi
source_ip: 10.10.1.7
dest_ip: 172.16.10.50
file_name: out_irl
------------
```

```
=== AI RECOMMENDATION ===
Threat category: Credential Compromise
Risk score: 30 Severity: Medium Confidence: 0.2

Phase-wise recommendations:
  Containment:
    1. Isolate affected host from network
    2. Force password reset and revoke sessions
    3. Block suspicious source IPs at perimeter
  Eradication:
    1. Search for indicators of compromise and remove backdoors
    2. Rotate credentials and reissue secrets
    3. Patch vulnerable services
  Recovery:
    1. Restore services and re-enable accounts with new credentials
    2. Monitor for re-occurrence
  Post-Incident:
    1. Perform root-cause analysis and update playbooks
    2. User awareness/training if phishing involved

Flat prioritized recommendations:
  1. Isolate affected host from network
  2. Force password reset and revoke sessions
  3. Search for indicators of compromise and remove backdoors
  4. Rotate credentials and reissue secrets
  5. Restore services and re-enable accounts with new credentials
  6. Monitor for re-occurrence
  7. Perform root-cause analysis and update playbooks
  8. User awareness/training if phishing involved
Explanation: Threat categorized as Credential Compromise. Heuristic risk score 30 suggests Medium se
verity.
```

- *Figure 3 & 4 — Excerpt from agent_results.json demonstrating saved, auditable analysis.*


```
{
  "result_id": "191df517-20bc-44be-b4b4-6e8ae8af3bb1",
  "alert": {
    "id": 4,
    "timestamp": "2025-11-14T19:17:16.336410+05:30",
    "alert_type": "suspicious login",
    "description": "I logged into my system this morning and found a new file , which wasnt c
    "host": "pc-20",
    "username": "Preethi",
    "source_ip": "10.10.1.7",
    "dest_ip": "172.16.10.50",
    "file_name": "out_irl"
  },
  "analysis": {
    "threat_category": "Credential Compromise",
    "risk_score": 30,
    "severity": "Medium",
    "confidence": 0.2,
    "phase_recommendations": {
      "Containment": [
        "Isolate affected host from network",
        "Force password reset and revoke sessions",
        "Block suspicious source IPs at perimeter"
      ],
```

```
      "Post-Incident": [
        "Perform root-cause analysis and update playbooks",
        "User awareness/training if phishing involved"
      ]
    },
    "recommendations": [
      "Isolate affected host from network",
      "Force password reset and revoke sessions",
      "Search for indicators of compromise and remove backdoors",
      "Rotate credentials and reissue secrets",
      "Restore services and re-enable accounts with new credentials",
      "Monitor for re-occurrence",
      "Perform root-cause analysis and update playbooks",
      "User awareness/training if phishing involved"
    ],
    "explanation": "Threat categorized as Credential Compromise. Heuristic risk score 30 sugge
  },
  "saved_by": "interactive_user",
  "timestamp_saved": "2025-11-14T13:47:37.374648"
}
```


---

5. Prompting & LLM integration (Future work)

The current prototype uses deterministic heuristics (mock LLM) for reproducibility and safety. For real LLM integration, the recommended approach:

1. Compute and pass the heuristic features (threat category, risk score, confidence) together with the raw alert JSON as context to the model.
2. Use a strict system/user prompt instructing the model to return **only JSON** in this schema:

```
{
 "threat_category":"Credential Compromise",
 "risk_score":75,
 "severity":"Critical",
 "confidence":0.85,
 "phase_recommendations":{ "Containment":[...], "Eradication":[...], ... },
 "recommendations":["..."],
 "explanation":"..."
}
```


3. Validate returned JSON, enforce schema checks, and **fallback** to heuristic recommendations if parsing fails or the model indicates low confidence.
3. Require analyst confirmation for any automated action — never allow an unverified model to perform network or host changes autonomously.

---

6. Evaluation & Limitations

Strengths

- Reproducible and auditable: deterministic heuristics ensure consistent outputs that can be inspected.
- Demonstrates how AI (or LLMs) can augment IR by prioritizing actions and mapping to IR phases.
- Interactive workflow supports human-in-the-loop decision making.

Limitations

- Mock LLM is not adaptive and cannot learn from new patterns.
- Real LLMs risk hallucinations; require strict prompts, validation, and guardrails.
- Prototype lacks live telemetry integrations (SIEM, EDR) and automated evidence collection required for real SOC operations.

---

7. Future Work

- Integrate OpenAI (or another LLM) with strict JSON output and schema validation.
- Build a Streamlit UI for a more user-friendly analyst experience (accept/reject buttons, playbook templates).

- Connect to SIEM/EDR to ingest logs and provide richer evidence in prompts.
- Add audit trails, role-based access, and approval workflows for any automated actions.

---

8. Conclusion

This mini-agent demonstrates a practical and safe path to AI-augmented incident response: combine deterministic heuristics, phase-mapped playbooks, and a human analyst in the loop. The prototype is easy to extend to LLMs and to integrate with SOC infrastructure while preserving safety and auditability.