**SOC-in-a-Box** project plan

Here’s a detailed project plan and project management plan for your **SOC-in-a-Box** project, designed to build a next-generation unified Security Operations Center leveraging open-source tools and the ELK stack.

**1. Project Overview**

**Project Title:**

SOC-in-a-Box: Building a Next-Generation Unified Security Operation Center

**Project Objective:**

To design and implement a cost-effective, scalable, and AI-enabled SOC platform using open-source tools for monitoring, detection, response, and threat intelligence against cyber threats.

**Key Components:**

* **Data Sources**: Elastic Endpoint, Syslog, Beats (Winlogbeat, Filebeat, etc.), Auditbeat, Metricbeat, threat feeds (MISP, OTX).
* **Data Processing**: Logstash, Elasticsearch.
* **Incident Response**: TheHive, Cortex XSOAR, Elastic SIEM.
* **AI and Machine Learning**: Kibana AI, Elastic ML models, UEBA, Generative AI.
* **Threat Simulation**: Infection Monkey, OpenVAS, Atomic Red Team.
* **Threat Intelligence**: MISP, SIGMA, YARA, Elastic rules databases.
* **Monitoring & Visualization**: Kibana dashboards.

**2. Project Plan**

**2.1. Milestones and Deliverables**

| **Phase** | **Timeline** | **Key Activities** | **Deliverables** |
| --- | --- | --- | --- |
| **Initiation** | Week 1-2 | Requirements gathering, stakeholder identification. | Project charter, initial scope document. |
| **Planning** | Week 3-4 | Define architecture, tools selection, resource planning. | Detailed project management plan. |
| **Environment Setup** | Week 5-7 | Install Ubuntu servers, configure Docker, Elastic Stack. | Baseline system ready for integration. |
| **Integration** | Week 8-10 | Integrate Elastic SIEM, TheHive, Cortex XSOAR, MISP. | Functional SOC components integrated. |
| **AI & Automation** | Week 11-14 | Configure Elastic ML, UEBA, Generative AI modules. | AI-enabled monitoring and automation enabled. |
| **Testing** | Week 15-17 | Simulate threats using Infection Monkey, Atomic Red Team. | Validated SOC performance metrics. |
| **Training** | Week 18-20 | Provide documentation and training for stakeholders. | SOC training manual, use case guides. |
| **Deployment** | Week 21 | Final deployment to production environment. | Fully operational SOC-in-a-Box platform. |
| **Maintenance** | Ongoing | Continuous monitoring, feedback incorporation. | Updates and improvement logs. |

**2.2. Work Breakdown Structure (WBS)**

**1. Requirements Gathering**

* Identify security challenges.
* Define stakeholders and their roles.

**2. Tool Selection**

* Evaluate ELK stack, TheHive, Cortex XSOAR, etc.
* Perform a compatibility analysis.

**3. System Design**

* Develop architecture diagrams.
* Specify data ingestion pipelines.

**4. Environment Setup**

* Set up VMs, install Ubuntu Server.
* Configure Docker for tools installation.

**5. Implementation**

* Deploy Elastic Stack and Fleet Server.
* Configure TheHive and Cortex XSOAR integrations.
* Enable threat intelligence via MISP.

**6. AI Enablement**

* Train Elastic ML models.
* Configure UEBA and automated response actions.

**7. Testing & Validation**

* Simulate APT scenarios using Infection Monkey and Atomic Red Team.
* Fine-tune detection rules and workflows.

**8. Training & Handover**

* Develop training modules for SOC operations.
* Conduct training workshops for analysts.

**3. Project Management Plan**

**3.1. Scope Management**

* Define boundaries for the SOC project.
* Ensure no scope creep by establishing strict change control processes.

**3.2. Time Management**

* Create a Gantt chart to monitor progress.
* Conduct weekly status meetings to review timelines.

**3.3. Cost Management**

* Use open-source tools to minimize software costs.
* Allocate budget for hardware and network infrastructure.

**3.4. Quality Management**

* Perform regular code and architecture reviews.
* Define KPIs such as detection rate, mean time to detect (MTTD), and mean time to respond (MTTR).

**3.5. Resource Management**

* Assign team roles: project manager, SOC analyst, threat intelligence lead, etc.
* Utilize cloud resources if local hardware is insufficient.

**3.6. Risk Management**

| **Risk** | **Mitigation** |
| --- | --- |
| Integration issues | Use Docker for modular setup and testing. |
| Resource shortages | Cross-train team members for flexibility. |
| Data overload | Implement scalable Elastic storage solutions. |

**3.7. Communication Plan**

* Weekly reports to stakeholders.
* Use collaboration tools like Microsoft Teams, Slack.

**3.8. Procurement Plan**

* Purchase hardware (servers, switches).
* Use community-supported open-source tools.

**4. Tools & Technologies**

* **Infrastructure**: VMware Workstation, Ubuntu Server.
* **Security Tools**: Elastic Stack, MISP, TheHive, Cortex XSOAR.
* **Development**: GitHub for version control, Jenkins for CI/CD.
* **Collaboration**: Microsoft Teams, Asana.