Project Scope Document for Celador:Automated Threat Intelligence Collection and Enrichment Platform

Project Title:

Celador - Automated Threat Intelligence Collection and Enrichment Platform

---

Project Overview:

Celador is an automated threat intelligence collection and enrichment platform designed to gather, enrich, and present real-time threat intelligence data from open-source feeds. The platform will pull threat data from selected free sources, enrich it with contextual information using APIs, and display the results in an intuitive dashboard for security professionals.

Objectives:

1. **Automate the collection** of threat intelligence data from multiple sources.
2. **Enrich the collected data** with additional information, such as geolocation, reputation scores, and service information.
3. **Present the enriched data** in an easy-to-understand format via a web-based dashboard.
4. Provide actionable insights for security professionals to identify and respond to network security threats more effectively.

Project Deliverables:

1. **Automated Data Collection**: Python scripts to pull data from threat intelligence feeds.
2. **Data Enrichment**: Additional context added using free APIs.
3. **Data Storage**: Store data in JSON or a lightweight database (SQLite).
4. **Visualization**: A web-based dashboard built using Grafana or Flask.
5. **Documentation**: Complete user guide and technical documentation for setting up and using the platform.

Key Features:

- **Threat Intelligence Collection**: Periodic collection of data from open-source feeds.
- **Data Enrichment**: Enrich the raw threat intelligence data with contextual information.
- **Dashboard**: Display the data in a user-friendly format, including threat types, enrichment details, and geolocation.
- **Automation**: Set up to collect and enrich data automatically at regular intervals.

Technology Stack:

1. **Programming Language**: Python (for data collection, enrichment, and automation)
2. **Data Storage**: JSON or SQLite (to store the collected and enriched data)
3. **Dashboard**: Grafana or Flask for displaying data

Step-by-Step Plan:

Step 1: Data Collection

Celador will collect threat intelligence from the following **three open-source threat intelligence feeds**:

1. **AbuseIPDB**: Provides reputation data on malicious IPs, known for abusive behavior.
    - **Endpoint**: IP blacklist and abuse reports.
    - **API**: Free tier API available.
2. **AlienVault OTX (Open Threat Exchange)**: Offers global threat intelligence and indicators of compromise (IOCs) such as IP addresses, domains, and malware.
    - **Endpoint**: Pulse data (IOCs).
    - **API**: Free API access available with registration.
3. **PhishTank**: Specializes in phishing domains and URLs.
    - **Endpoint**: Blacklisted phishing URLs.
    - **API**: Provides an API to retrieve phishing data for free.

Step 2: Data Enrichment

Once the data is collected, Celador will enrich it using the following **three free enrichment APIs**:

1. **VirusTotal**: Enriches the collected data with reputation scores and historical analysis for domains, IP addresses, and files.
    - **Purpose**: Check for known malware or malicious behavior associated with the indicators (IPs, URLs).
    - **API**: Free API with limited requests per minute.
2. **Shodan**: Provides information about open ports and services running on a specific IP, making it useful for understanding the potential attack surface of a malicious IP.

- **Purpose**: Get data about services running on the identified IPs.
- **API**: Free API tier available with rate limits.

3. **IPinfo**: Offers geolocation and ASN (Autonomous System Number) details about IP addresses.
    - **Purpose**: Add geolocation data and ISP information to the collected IP addresses.
    - **API**: Free API tier for basic IP lookups.

Step 3: Data Storage

The collected and enriched threat intelligence will be stored locally:

- **Storage Method**: Initially, the data will be stored in **JSON** format for simple prototyping and ease of access. For more structured storage, **SQLite** will be used.

Step 4: Visualization and Dashboard

Celador will feature a dashboard to visualize and display the enriched threat intelligence data:

1. **Dashboard Platform**:
    - **Grafana**: A widely-used open-source platform that integrates easily with various data sources like JSON and SQLite.
    - Alternatively, **Flask** can be used for building a custom dashboard if more control is needed.
2. **Visual Elements**:
    - **Threat Types**: Breakdown of malicious IPs, domains, and URLs by threat category.
    - **Geolocation Map**: A visual representation of the geolocation of malicious IP addresses.
    - **Enrichment Details**: Additional context, such as reputation scores from VirusTotal, open ports/services from Shodan, and geolocation from IPinfo.

Step 5: Automation

Celador will automate the entire process to run at scheduled intervals:

- **Scheduling**: Use **cron jobs** (Linux) or **Task Scheduler** (Windows) to automate data collection and enrichment at predefined intervals (e.g., daily or hourly).

Timeline:

Day 1-2: Project Setup and Environment Configuration

- Set up the development environment (install Python, necessary libraries like requests, Flask, or Grafana).
- Create a GitHub repository for version control and documentation.

Day 3-4: Threat Intelligence Collection (AbuseIPDB)

- Write the Python script to collect data from **AbuseIPDB**.
- Test the script to ensure correct API interaction and data collection.

Day 5-6: Threat Intelligence Collection (AlienVault OTX and PhishTank)

- Write Python scripts for collecting threat data from **AlienVault OTX** and **PhishTank**.
- Test both APIs and ensure all data sources are working as expected.

Day 7-8: Data Enrichment (VirusTotal)

- Implement the enrichment process using the **VirusTotal** API to check reputation and threat details for collected IPs, domains, and URLs.
- Test with real data and store the results in a local JSON file.

Day 9: Data Enrichment (Shodan and IPinfo)

- Integrate the **Shodan** and **IPinfo** APIs to enrich data with open service information and geolocation details.
- Test the enrichment process for all collected data.

Day 10: Data Storage (SQLite or JSON)

- Set up local data storage in **SQLite** or **JSON** to store the collected and enriched data for easier retrieval and visualization.
- Test basic queries or file access to ensure data is properly stored.

Day 11-12: Dashboard Setup (Grafana or Flask)

- Install and configure **Grafana** (or set up **Flask**) for the dashboard.
- Create basic visualizations (e.g., geolocation map, IP reputation breakdown, service information).
- Ensure that the data is displayed correctly in the dashboard.

Day 13: Automation with Cron Jobs/Task Scheduler

- Set up cron jobs (Linux) or Task Scheduler (Windows) to automate the collection and enrichment process on a regular basis (e.g., daily or hourly).
- Test the automation to ensure the data is being collected and enriched as scheduled.

Day 14: Testing and Final Touches

- Perform end-to-end testing to verify that data is being collected, enriched, stored, and visualized correctly.
- Write basic documentation (installation guide, usage guide) and finalize the project for publishing on GitHub.

Documentation & Publication:

Once the project is completed, it will be shared with the public through:

1. **GitHub**: Publish the code, scripts, and documentation on GitHub.
2. **Blog Post**: Write a blog post or tutorial explaining the project setup, key features, and use cases for the platform.