System Architecture and Security Design

AUSAINT - OSINT Tool

Submitted By:

Muhammad Umar Maqsood
Shamina Durrani
Afnan bin Abbas
Aqib Shakeel

March 21, 2025

1 Introduction

The AUSAINT OSINT Investigative Suite is a secure, scalable, and efficient intelligence-gathering tool designed to assist cybersecurity professionals and researchers in obtaining publicly available intelligence while ensuring data integrity, confidentiality, and availability. This document outlines the system architecture, security controls, and secure design measures integrated into the system.

2 System Architecture

2.1 High-Level Architecture

The system follows a multi-layered architecture, ensuring modularity, security, and performance. The main components are:

- **Client Layer:** The user-facing frontend developed using React.js to provide an intuitive UI for OSINT queries and intelligence visualization.
- **API Gateway & Backend Services:** The Flask-based backend processes OSINT requests, enforces authentication, and interacts with data sources.
- **OSINT Data Retrieval Modules:** These modules use open-source intelligence tools such as Sherlock, Twint, Sublist3r, and WHOIS to fetch data.
- **AI-Powered Threat Analysis:** Utilizes machine learning models to detect anomalies, classify threats, and score risks.
- **Database Layer:** A PostgreSQL database stores query logs, user information, and collected intelligence data with encryption.
- **Security Layer:** Comprising authentication, encryption, access control, and real-time monitoring mechanisms to safeguard the platform.

2.2 System Architecture Diagram

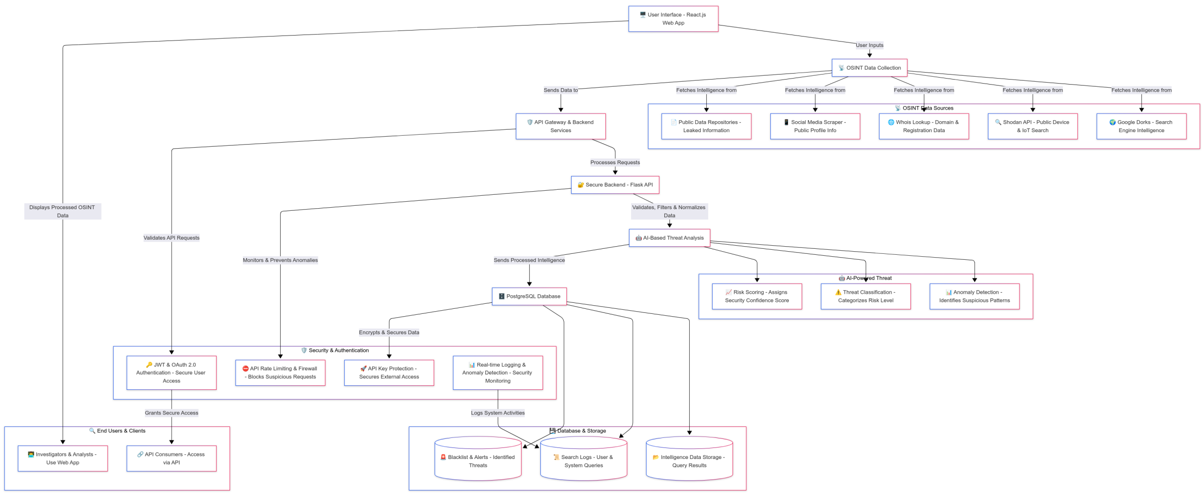
🖥️ User Interface - React.js Web App
User Inputs
📡 OSINT Data Collection
Sends Data to
Fetches Intelligence from
Fetches Intelligence from
Fetches Intelligence from
Fetches Intelligence from
Fetches Intelligence from
📡 OSINT Data Sources
🛡️ API Gateway & Backend Services
📄 Public Data Repositories - Leaked Information
📱 Social Media Scraper - Public Profile Info
🌐 Whois Lookup - Domain & Registration Data
🔍 Shodan API - Public Device & IoT Search
🌍 Google Dorks - Search Engine Intelligence
Processes Requests
🔐 Secure Backend - Flask API
Validates, Filters & Normalizes Data
Displays Processed OSINT Data
🤖 AI-Based Threat Analysis
Validates API Requests
Monitors & Prevents Anomalies
Sends Processed Intelligence
🤖 AI-Powered Threat
📈 Risk Scoring - Assigns Security Confidence Score
⚠️ Threat Classification - Categorizes Risk Level
📊 Anomaly Detection - Identifies Suspicious Patterns
🗄️ PostgreSQL Database
Encrypts & Secures Data
🛡️ Security & Authentication
🔑 JWT & OAuth 2.0 Authentication - Secure User Access
⛔ API Rate Limiting & Firewall - Blocks Suspicious Requests
🚀 API Key Protection - Secures External Access
📊 Real-time Logging & Anomaly Detection - Security Monitoring
Grants Secure Access
Logs System Activities
🔍 End Users & Clients
👨‍💻 Investigators & Analysts - Use Web App
🔗 API Consumers - Access via API
💾 Database & Storage
🚨 Blacklist & Alerts - Identified Threats
📜 Search Logs - User & System Queries
📂 Intelligence Data Storage - Query Results

Figure 1: System Architecture Diagram of AUSAINT OSINT Tool

3 Security Controls

3.1 Authentication & Access Control

To ensure secure user interactions:

- **JWT-based Authentication:** Secure, stateless authentication tokens.
- **Role-Based Access Control (RBAC):** Restricts unauthorized access to sensitive OSINT functions.
- **Session Expiry & Re-authentication:** Reduces risk of unauthorized access via session hijacking.

3.2 Encryption & Secure Communication

Ensuring data protection during transmission and storage:

- **TLS 1.2+ Encryption:** All client-server communication is encrypted.
- **AES-256 Encryption for Data Storage:** Sensitive OSINT logs and queries are securely stored.
- **API Key Protection:** API keys are managed securely to prevent unauthorized use.

3.3 Rate Limiting & Intrusion Detection

Preventing abuse and monitoring threats:

- **API Rate Limiting:** Limits excessive requests to prevent DoS attacks.
- **Intrusion Detection System (IDS):** Detects anomalous activity and blocks potential threats.
- **Real-time Logging & Alerts:** Tracks failed authentication attempts and suspicious activity.

4 Secure Design Measures

4.1 Secure API Development

The following principles are applied to ensure secure API communication:

- **Input Validation & Sanitization:** Protects against SQL Injection and XSS attacks.
- **Restricted CORS Policy:** Only trusted domains are allowed to access the API.
- **Logging & Audit Trails:** Maintains detailed logs for accountability.

4.2 Secure Deployment & Monitoring

To maintain long-term security and operational stability:

- **Docker Containerization:** Isolates services, reducing cross-service vulnerabilities.
- **Continuous Security Audits:** Regular penetration testing and vulnerability assessments.
- **Automated Patch Management:** Ensures up-to-date security measures are applied.

4.3 Secure Design Diagram

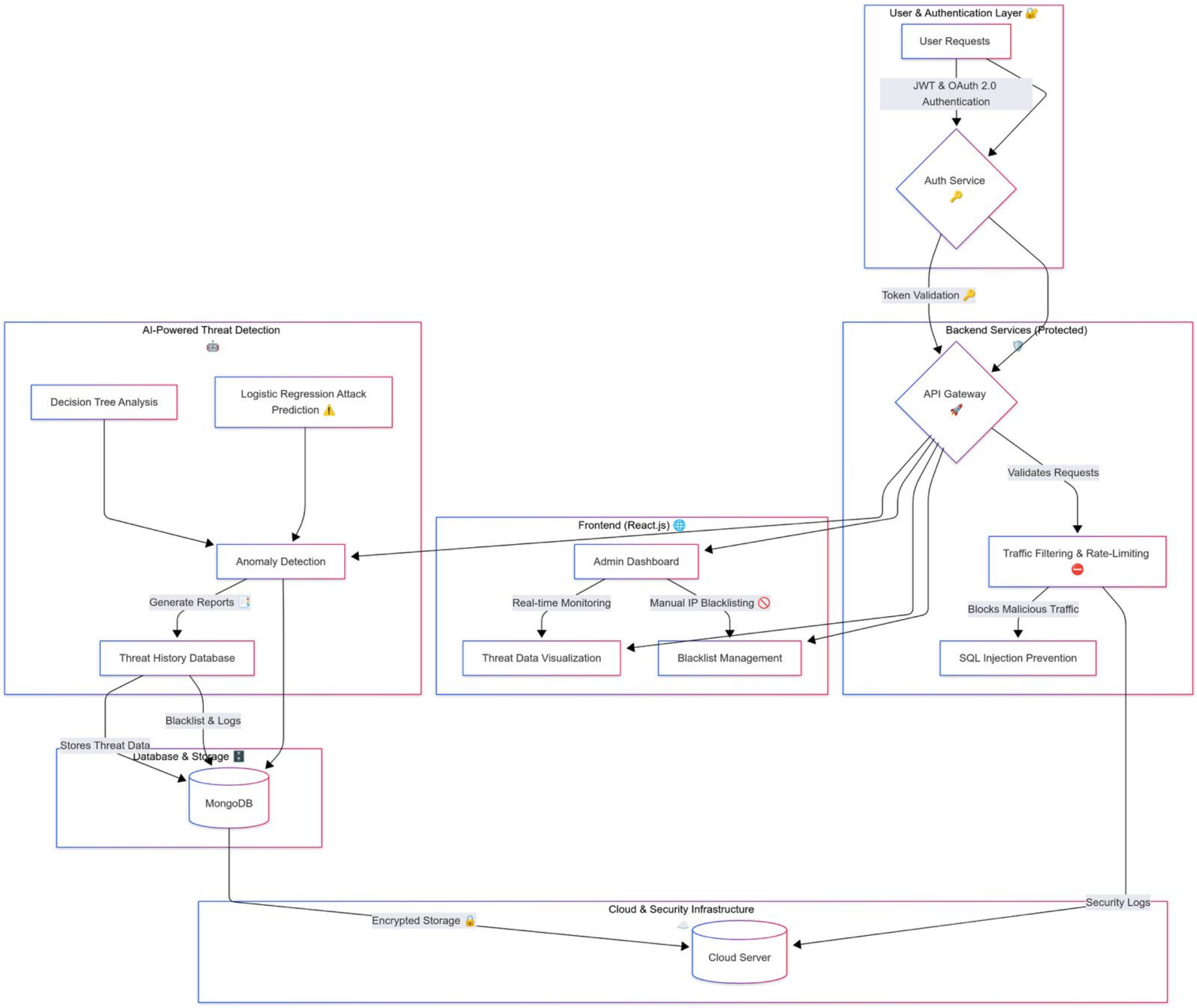

Figure 2: Secure Design Architecture of AUSAINT OSINT Tool

5 Conclusion

The AUSAINT OSINT Tool is developed with security-first principles, ensuring safe and ethical intelligence gathering while maintaining strong access control, encryption, and monitoring mechanisms. By implementing best practices in secure software development, AUSAINT provides a robust framework for open-source intelligence operations.