4. Wazuh-AI Processing Pipeline

The Wazuh-AI processing pipeline is a critical component designed to automate the analysis and enrichment of security alerts from Wazuh. It processes alerts, enriches them using AI, and stores the results in the PostgreSQL database. This section outlines the conceptual flow and key considerations for this pipeline.

Overview

This pipeline aims to transform raw Wazuh alerts into actionable intelligence. The core process involves:

1. **Watcher Functionality**: A script continuously monitors Wazuh alerts.
2. **AI Integration**: An AI component processes these alerts, generating enriched data, often in SQL format.
3. **Webhook-Rundeck Connection**: A webhook triggers Rundeck jobs based on new alerts, orchestrating the AI processing and subsequent actions.
4. **Database Integration**: The enriched data is then pushed into the PostgreSQL database for storage and further analysis.

Key Considerations and Future Improvements

During the prototype phase, several areas were identified for improvement, which are crucial for a robust production-ready system. Addressing these points will enhance the pipeline's reliability, efficiency, and data quality:

- **JSON Output Refinement**: The JSON output from the AI component needs to be consistently clean, structured, and append-only. Current issues include overwriting data and overly detailed or messy formatting. Implement robust data serialization and deserialization mechanisms.
- **Filtering Mechanism Tuning**: The alert filtering mechanism, intended to process only higher-risk alerts (e.g., level 4+), was inconsistent. This requires fine-tuning to ensure only relevant and actionable alerts trigger the AI processing, reducing noise and improving efficiency.

- **Rundeck Permission Management**: Persistent file permission issues within Rundeck prevented consistent AI script execution. Implement proper user and file permissions to ensure Rundeck can reliably access and execute necessary scripts.
- **Watcher Automation**: The watcher script currently requires manual initiation. Configure the watcher to start automatically on system boot (e.g., using `systemd` services) to ensure continuous monitoring and processing.