zenatix

Assignment for Architecture Design of Edge Alarms

Background

You are working with a Linux-based IoT Gateway. This gateway interfaces with various devices through multiple protocols such as BACnet, Modbus, WiFi, and LoRA. These devices gather different types of data (e.g., Temperature, Humidity, CO2, Energy, etc.) and make it available via REST API or MQTT locally.

Objective

Design an Alarms Service that can:

- Read data from MQTT local endpoints.
- Compute alarms based on predefined rules.
- Publish the alarm back on http/mqtt endpoint.
- Support parallel execution of multiple processes.

Alarm Types

1. **Simple Threshold Violation Alarms**: Triggered when a data point exceeds a certain threshold for a predefined duration. E.g., Temperature > 24°C for 60 minutes.
2. **Conditional Threshold Violation Alarms**: Triggered based on a primary condition and an additional shunt condition. E.g., Temperature > 24°C for 60 minutes AND Current > 0 (indicating the device is on), with the threshold data potentially coming from another device.

Architectural Considerations

1. **Stateful Application**:
    - The service must maintain its state across reboots of the application or the gateway.
    - Consider using persistent storage to record the state and progress of alarm conditions.
2. **Data Model for Alarm Rules**:
    - Define a flexible yet structured data model that can accommodate various alarm types and conditions.
    - Include attributes like sensor ID, threshold values, duration, shunt conditions, etc.
3. **Storage Medium for Alarm Rules**:
    - Select a storage solution suitable for the gateway's hardware and software environment.
    - Consider lightweight databases like SQLite or file-based storage for ease of integration and maintenance.
4. **Alarm Processing Logic**:
    - Develop a robust mechanism to evaluate data against the defined alarm rules.
    - Implement efficient data handling to minimize latency and resource usage.
5. **Scalability and Extendability**:
    - Ensure the design can handle an increasing number of devices and alarm rules.
    - Design with modularity in mind to facilitate future extensions, such as new types of alarms or integration with external systems.
6. **User Interface and Configuration**:

    - Provide a user-friendly interface for configuring alarm rules.
    - Include options for viewing active alarms and historical alarm data.
7. **Publish Alarm**
    - Publish Alarms back on MQTT
8. **Logging and Monitoring**:
    - Implement comprehensive logging for troubleshooting and performance monitoring.
    - Consider integration with existing monitoring tools for real-time insights.

Deliverables

- A detailed architectural diagram illustrating the components and data flow. If you would use some third party (open source) service to implement it then mention why that service is most suitable and what changes/configurations will be required on top of that service.
- A specification document outlining the assumptions made (beyond what has been explained in the assignment), corner cases addressed (whatever you can think of based on your past experience), data models, storage format, and processing logic.
- A prototype or proof of concept demonstrating key functionalities (either running on your computer or on some edge platform like Raspberry Pi) programmed in Python.

Mode of Submission of the assignment

- A MS word document
- Source code of prototype as .zip file