**Log Intelligence System - Design Document**

**1. Introduction**

**1.1 Purpose**

The purpose of this document is to provide a detailed design description of the Log Intelligence System (LIS). This document serves as a guide for developers and stakeholders to understand the architecture and design of the system.

**1.2 Scope**

The Log Intelligence System aims to proactively monitor logs using AI/ML to filter critical logs and provide real-time notifications and insights via dashboards. The system will integrate with AWS services for log ingestion, processing, alerting, visualization, and security monitoring.

**1.3 Definitions, Acronyms, and Abbreviations**

* **LIS**: Log Intelligence System
* **AWS**: Amazon Web Services
* **NLP**: Natural Language Processing
* **SNS**: Simple Notification Service
* **IAM**: Identity and Access Management
* **GuardDuty**: AWS Threat Detection Service

**2. Overall Description**

**2.1 Product Perspective**

The Log Intelligence System will be deployed in AWS and will interact with log sources to collect, analyze, and provide insights. The system will function as a cloud-based solution with real-time processing, alerting, and security threat detection capabilities.

**2.2 Product Functions**

1. **Log Collection** – Logs are collected from various sources (Windows, Linux, applications, firewalls, routers, switches).
2. **Log Processing** – Logs are analyzed and filtered based on severity using AWS AI/ML services.
3. **Log Storage** – Logs are stored in **Amazon S3** and indexed in **Amazon OpenSearch**.
4. **Alerts & Notifications** – Critical logs trigger alerts via **SNS (Email/SMS/Teams)**.
5. **Visualization** – Real-time dashboards are generated using **Amazon QuickSight**.
6. **Security Monitoring** – Continuous security threat detection with **AWS GuardDuty**.

**2.3 User Classes and Characteristics**

* **IT Administrators**: Monitor logs and investigate critical issues.
* **Security Analysts**: Analyze logs for security threats.
* **System Engineers**: Maintain log pipeline and infrastructure.

**2.4 Operating Environment**

* Cloud-based deployment on AWS.
* Web-based dashboard using **Amazon QuickSight**.
* Multi-region availability for redundancy.

**3. System Architecture**

**3.1 Architecture Overview**

* **Data Sources**: Windows/Linux servers, applications, network devices, firewalls.
* **Log Collection**: Log server consolidates logs (SysLogs, application logs, infrastructure logs).
* **Ingestion**: Logs are ingested via **Amazon Kinesis Data Firehose or Using Python Script** to inject into **Amazon S3**.
* **Processing**: **AWS Lambda** triggers AI/ML-based log filtering using **AWS Bedrock**.
* **Storage & Search**: Logs are stored in **Amazon S3** and indexed in **AWS OpenSearch**.
* **Security Monitoring**: **AWS GuardDuty** detect threats.
* **Alerting**: **Amazon SNS**, **Amazon Chatbot**, and **Amazon EventBridge** send notifications.
* **Visualization**: Dashboards are created using **Amazon QuickSight**.

**3.2 Component Details**

* **Log Server**: Aggregates logs from multiple sources.
* **Amazon Kinesis Firehose**: Streams logs to S3 for storage.
* **AWS Lambda**: Processes logs and applies AI-based classification.
* **AWS Bedrock**: Utilizes AI/ML for log intelligence and anomaly detection.
* **Amazon OpenSearch**: Enables log search, indexing, and analytics.
* **AWS GuardDuty**: Detects threats and suspicious activities in logs.
* **Amazon SNS**: Sends alerts for critical logs.
* **Amazon Chatbot**: Integrates with **Microsoft Teams** for notifications.
* **Amazon QuickSight**: Visualizes log data for real-time monitoring.
* **Amazon EventBridge**: Automates actions based on log severity.

**4. Detailed Design**

**4.1 Data Model**

**4.1.1 Log Storage Structure**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Type** | **Description** |
| LogID (PK) | String | Unique identifier for each log entry |
| Timestamp | DateTime | Time when log entry was recorded |
| Source | String | Device or application generating the log |
| Severity | Enum | Severity level (Critical, High, Warning, Info) |
| Message | String | Log message description |
| ProcessedStatus | Boolean | Indicates whether the log has been processed by AI |

**4.2 User Interface Design**

**4.2.1 Dashboard Features**

* **Log Search & Filtering**: Search logs based on severity, timestamp, and source.
* **Severity-Based Categorization**: Critical logs highlighted for quick action.
* **Real-Time Alerts & Visualizations**: Dashboard updates dynamically based on new log data.

**4.3 API Design**

**4.3.1 API Endpoints**

* **POST /logs**: Ingest logs into S3
* **GET /logs**: Retrieve filtered logs
* **POST /alerts**: Send notifications for critical logs

**5. Security and Privacy**

* **AWS IAM** for role-based access control (RBAC).
* **AWS GuardDuty** for real-time threat detection.
* **Encryption** of logs in transit and at rest using **AWS KMS**.
* **Multi-Factor Authentication (MFA)** for user access to dashboards.

**6. Testing Strategy**

* **Unit Testing**: Validate AWS Lambda functions for log processing.
* **Integration Testing**: Simulate real-time log ingestion and AI-based classification.
* **Security Testing**: Verify AWS GuardDuty alerts for security anomalies.
* **User Acceptance Testing (UAT)**: End-user validation for log filtering, alerts, and dashboards.