**System Health and Anomaly Detection System Documentation**

**Introduction**

The System Health and Anomaly Detection System is designed to fetch data from Splunk, detect anomalies using machine learning, generate system health reports, and integrate Cyber Threat Intelligence (CTI) to provide security suggestions. This document outlines the functionalities, how to adapt them to your system, and suggestions for further development.

**Prerequisites or Requirements**

Before executing the System Health and Anomaly Detection script, ensure the following prerequisites are met:

1. **Access to Splunk:** Obtain valid credentials (username and password) for Splunk to retrieve relevant data. Ensure the provided access privileges include the necessary permissions to fetch data from the specified indexes.
2. **Python Environment:** The script requires a Python environment to execute. Ensure Python 3.x is installed on your system.
3. **Required Python Packages:** Install necessary Python packages. Execute the following command to install required packages using pip:

*pip install pandas scikit-learn splunk-sdk requests*

1. **Network Connectivity:** Ensure stable network connectivity to access Splunk repositories and external CTI sources.
2. **Modification of Placeholder Information:** Modify placeholder information such as 'username', 'password', 'your\_index', 'search\_criteria', and other placeholders in the script to match your system configurations, credentials, and specific search criteria.
3. **Security Considerations:** Implement proper security measures to secure sensitive information, especially when handling authentication details or integrating external data sources.

**Code Overview**

The code comprises several functions, each serving a specific purpose. Additionally, several Python modules are imported to support various functionalities:

**Module Imports**

The following Python modules are imported in the code to enable specific functionalities:

* **pandas (import pandas as pd):** Pandas is used to handle data in tabular form and perform data manipulation tasks.
* **sklearn IsolationForest (from sklearn.ensemble import IsolationForest):** The Isolation Forest algorithm is employed for anomaly detection, allowing the identification of potential irregularities within the system's data.
* **splunklib (from splunklib.client import Service):** This library facilitates interaction with Splunk, enabling data retrieval from Splunk repositories.
* **requests (import requests):** The 'requests' library is used for making HTTP requests, particularly to fetch Cyber Threat Intelligence data from external sources or APIs.
* **datetime (from datetime import datetime, timedelta):** The 'datetime' module provides functionality to handle dates and time, useful for defining time intervals for data retrieval and analysis.
* **logging (import logging):** The 'logging' module is used for error handling and logging activities, aiding in tracking errors and debugging information throughout the code execution.

**Functionality Overview**

The code comprises several functions, each serving a specific purpose:

**1. fetch\_data\_from\_splunk(start\_time, end\_time)**

This function retrieves data from Splunk using the Splunk SDK.

* **Modification:** Replace placeholders with your Splunk credentials, search criteria, and index information.

**2. preprocess\_data(raw\_data)**

Preprocesses raw data fetched from Splunk.

* **Modification:** Implement data preprocessing specific to your system.

**3. train\_anomaly\_model(data)**

Trains an Isolation Forest anomaly detection model.

* **Modification:** Adjust model parameters based on your anomaly detection requirements.
* **Security Consideration:** Ensure data security during training and model deployment to prevent unauthorised access to the model or training data.

**4. suggest\_alerts(anomalies)**

Suggests potential alerts based on detected anomalies.

* **Modification:** Define thresholds or criteria for alert suggestions.
* **Security Consideration:** Handle alerts securely to avoid exposure of sensitive information in potential alerts.

**5. review\_alerts(alerts\_to\_review)**

Reviews and approves suggested alerts by a cybersecurity engineer.

* **Modification:** Modify the review process or user interaction to suit your workflow.
* **Security Consideration:** Ensure a secure approval process to prevent unauthorised modifications to the alerts.

**6. analyse\_patch\_levels(data)**

Analyses patch levels and system health.

* **Modification:** Implement analysis of patch levels, vulnerabilities, or system health in your system.
* **Security Consideration:** Ensure secure data analysis to prevent potential vulnerabilities during the analysis process.

**7. suggest\_investigation(data)**

Suggests areas for cybersecurity investigation based on system logs or events.

* **Modification:** Analyse data to suggest investigation areas based on your system's context.
* **Security Consideration:** Perform secure investigation suggestions to avoid unintended risks to the system.

**8. weekly\_changes\_and\_trends(data)**

Reports on weekly changes and trends observed in the system.

* **Modification:** Analyse system data to report on trends specific to your system.
* **Security Consideration:** Ensure data privacy and secure reporting to prevent unauthorised access to the system's trend information.

**9. fetch\_cti\_data()**

Fetches Cyber Threat Intelligence (CTI) data from an external source or API.

* **Modification:** Specify the URL, API endpoints, and authentication details for your CTI source.
* **Free Resources:** In case of unavailable CTI capabilities, organisations can use free resources like the Open Threat Exchange (OTX) by AlienVault or the National Vulnerability Database (NVD) for threat intelligence information.

**10. analyse\_cti\_data(cti\_data)**

Analyses CTI data to provide network security suggestions.

* **Modification:** Analyse CTI data to extract relevant threat intelligence and tailor security suggestions.
* **Free Resources:** Incorporate open-source threat intelligence feeds such as MISP (Malware Information Sharing Platform) or abuse.ch for additional threat intelligence.

**11. update\_report\_with\_cti(system\_report)**

Updates the system report with CTI-based security suggestions.

* **Modification:** Integrate CTI-based suggestions into the system report to enhance security insights.
* **Security Consideration:** Ensure secure integration of CTI information to prevent tampering or unauthorised access to the system report.

**12. generate\_system\_report()**

Generates a comprehensive system health and security report.

* **Modification:** Implement functions and variables as required to form the report specific to your system.
* **Security Consideration:** Ensure the confidentiality and integrity of the generated system report to prevent unauthorised modifications or access.