IDS AI Tool

System Description

Abstract

This is the System Description (SysD document) for the ”AI Tool” System according to the Eclipse Arrowehad documentation structure.

Contents

1 Overview

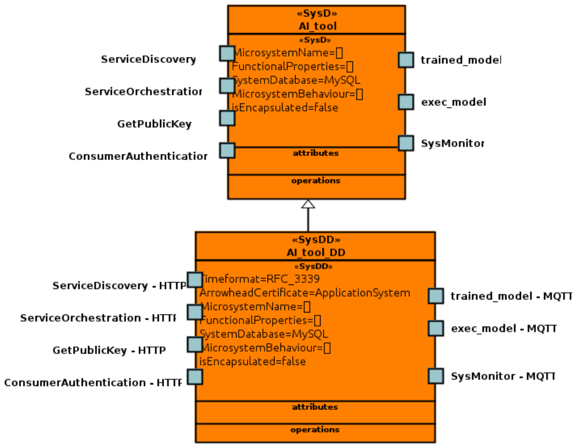

Figure 1: Block Diagram representation of the ”AI Tool” microsystem

The rest of this document is organized as follows. In Section 1.1, we reference major prior art capabilities of the system. In Section 1.2, we describe the intended usage of the system. In Section 1.3, we describe fundamental properties provided by the system. In Section 1.4, we describe delimitations of capabilities ofn the system. In Section 2, we describe the abstract service functions consumed or produced by the system. In Section 3, we describe the security capabilities of the system.

1.1 Significant Prior Art

The artificial intelligence-based intrusion detection system (IDS) at the edge builds on several significant previous works in the areas of network security, artificial intelligence and edge computing. Some of the most relevant work that has contributed to the development of this system is described below:

- **Traditional Intrusion Detection Systems:** Traditional IDSs, such as Snort and Suricata, have laid the groundwork for detecting malicious traffic patterns on networks. These systems use predefined signatures and rules to identify known threats.
- **Artificial Intelligence in Network Security:** The application of artificial intelligence techniques, such as machine learning and deep learning, has enabled the detection of advanced and unknown threats. Works such as "Deep Learning for Cyber Security Intrusion Detection"[1] has demonstrated the effectiveness of these techniques in identifying anomalous patterns in network traffic.
- **Edge Computing:** Edge computing enables data processing close to the source of generation, reducing latency and the bandwidth required to transmit data to the cloud.
- **Integration of AI and Edge Computing:** The combination of artificial intelligence and edge computing has been explored in studies that have shown how AI at the edge can improve the efficiency and responsiveness of security systems.

1.2 How This System Is Meant to Be Used

This system is used to detect an intrusion in a stream of data. It can get the pretrained model and the stream of data as input. The system will execute the model on the stream of data and detect an intrusion based on the signatures and on the anomalies. If an intrusion is detected, the system will send an alert to the alert system and to the network sensors system.

1.3 System functionalities and properties

1.3.1 Functional properties of the system

- The system can detect an intrusion in a stream of data
- The system can execute a pretrained model on a dataset
- The system can send an notification to the Alert system

1.3.2 Configuration of system properties

1.3.3 Data stored by the system

The model is stored as a tensor with the weights and the biases. The packets are stored in a buffer just the time to be executed by the model, and the results are stored in a buffer to be sent to the detection system.

1.3.4 Non functional properties

In order to reduce the energy consumption and keep the real time response, the model is reduced to be as small as possible.

1.3.5 Stateful or stateless

Stateless: the data is buffered and it is not kept by the system.

1.4 Important Delimitations

The system can be delimited by the computing power of the CPU/GPU. With light models and high-performance runtime for on-device AI such as LiteRT this can be solved. Nevertheless, reducing the size of the model may be a challenge and a field for research in order to have and accurate detection of intrusion.

2 Services

2.1 Produced service

- ”exec model”: The service determines if there was an intrusion in the stream it received and sends a boolean if there was to the alert system and the network sensors system.
- ”SysMonitor”: This service provides real-time log messages to an external monitoring system. It is based on the specifications detailed in the MicrosystemMonitor SD document.

2.2 Consumed services

- ”trained model”: The service receives the tensor of the model sent by the Training system and stores it in order to execute it.
- ServiceDiscovery: Essential for communication with the registry.
- ServiceOrchestration: Coordinates the system.
- GetPublicKey: Provides the necessary authorization mechanisms.
- ConsumerAuthentication: Manages the authentication of external entities wishing to access service data.

3 Security

- The AI-tool system utilizes secure protocols such as:
  - TLS (Transport Layer Security): Provides secure communication over the network, protecting data integrity and privacy during transmission.
  - MQTT over TLS: For low-latency, lightweight communication in IoT and edge environments, ensuring data is transmitted securely.
  - HTTP/HTTPS: For web-based communication, with HTTPS ensuring secure communication via encryption.

For Arrowhead certificate profile see github.com/eclipse-arrowhead/documentation

4 References

[1] M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, "Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study," *Journal of Information Security and Applications*, vol. 50, p. 102419, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214212619305046

5 Revision History

5.1 Amendments

Revision history and Quality assurance as per examples below

No.	Date	Version	Subject of Amendments	Author
1	2020-12-05	X.Y.Z		Tanyi Szvetlin
2	2021-07-14	X.Y.Z	Minor updates	Jerker Delsing
3	2022-01-12	X.Y.Z	Minor updates	Jerker Delsing

5.2 Quality Assurance

No.	Date	Version	Approved by
1	2022-01-10	X.Y.Z	