CEng 491 -- Project KickOff Document

Deep Learning based IOT Network Attack Detection System

Project acronym: DIONA

Description

Since the number of IoT devices have been increasing dramatically owing to improvements on Cloud and Fog computing, vulnerabilities to the cyber attacks and data breaches of enterprises, companies, etc. are also increased enormously. The connection between the IoT end-points and central control unit provides several functionalities. Having these functionalities and increasing use of Deep Learning in cyber security make the problem solvable by Deep Learning methods. Furthermore, there are no static IoT devices set-up for different scenarios. DIONA aims to solve this problem by closely observing the network traffic between the IoT devices and the central control unit with DL methods. These observation processes make DIONA a reliable solution to the cyber security problems mentioned above, since it provides a two-layered security pattern by its nature. The end-product will be able to work with different kinds of IoT end-points, since its training data is independent of the Network but dependent on the device itself. It will classify the traffic and label the corresponding device, and if it recognizes a threat, it will isolate the device from the Network. Since DIONA adaptable to all kinds of IoT devices, there is no limitation on the expected user group.

Master feature list

MF-1: A Unified Command and Control System will be implemented for managing and monitoring of IOT instances and their network.(Optional)
MF-2: Overall network traffic will be encrypted.
MF-3: System will utilize an indexed database system (search engine).
MF-4: Users will be able to observe the status via the front-end web server.
MF-5: DL will detect behavioral anomalies of IOT instances by analyzing the normal behavior of IOT instances.
MF-6: DL will detect malicious behavior on the network side.
MF-7: System will have a managed firewall.
MF-8: Logs can be pushed and pulled between the IOT devices, and parsed for the overall system usage.
MF-9: Security alerts for different scenarios will be generated.
MF-10: The network traffic will be observed by network sniffing methods.
MF-11: By the usage of a Rule-Based Detection Engine, the system will be able to monitor network traffic and generate alerts for possible network intrusions.
MF-12: The Log Parser and Unifier will parse, and unify if required, generated logs from the IoT Swarm and feed them to the DL Engine.

Work Packages

WP #	Term	WP title	Estimated number of person-months
1	491	Project planning and architecture design	3
2	491	Architecture implementation, control system preparation and database model decision	6
3	491	Implementation of LPUS (Log Parser & Unifier System)	6
4	491	Implementation of DL Engine for behavior analysis	8
5	492	Implementation of Security layer	6
6	492	Integration of DL Engine and Security appliance	7
7	492	Local testing of DL Engine Attack Detection system with IOT device actions	6
		Total:	**42**

Detailed Descriptions of High-Level Work Packages

WP1 - Project planning and architecture design

In this work package, the following functionalities / features / work items will be implemented

1. Develop the list of master features of the project. (All MFs)
2. Produce project development plan in accordance with the Master Feature List. (All MFs)
3. Design the overall architecture of the project. (All MFs)
4. Analyze risks and make a management plan. (All MFs)
5. Decide on a database model for behavioral classification and action logging. (MF-4)
6. Design and implement an architecture which will be the base for Deep Learning based analyzer, Network controller and Security implementation.

WP2 - Architecture implementation, control system preparation and database model

1. Setting up the fog layer infrastructure
2. Setting up the security layer infrastructure
3. Setting up the DL engine environment.

> Related MFs: MF-1, MF-4, MF-5 (extra MFs: MF-2)

WP3 - Implementation of LPUS (Log Parser & Unifier System)

1. Setting up an IoT Log Parser.
2. Setting up an IoT Log Unifier.
3. Preparing unified log files from datasets gathered from sources.
4. Indexing log files via IoT devices.
5. Preparing some generic categorizations and preparing templates which are info and status models for IOTs.

> Related MFs: MF-11, MF-12

WP4 - Implementation of DL Engine for behavior analysis

1. Do research about the best DL model for anomaly detection.
2. Decide on DL framework (PyTorch, TensorFlow, Keras, etc.)
3. Search for suitable datasets for the DL model.
4. Data pre-processing
5. Feature engineering
6. Training the models with normal behaviors
7. Analyze and classify abnormal situations.
8. Try the suitable DL models and compare the success rates.
9. Choose the optimal model that performs the best

> Related MFs: MF-5, MF-6, MF-12

WP5 - Implementation of Security layer

1. Setting up An isolated network for the security layer.
2. Installing log channels.
3. Installing packet sniffer.
4. Installing Parsing logs.
5. Installing monitoring systems.
6. Installing A network firewall.
7. Setting up rules for Security alerts.
8. Encrypting all traffic between the IoT fog and the security layer.
9. Encrypting all traffic between the security layer and the DL engine.

> Related MFs: MF-9, MF-10, MF-11

WP6 - Integration of DL Engine and Security layer

1. Setting up API between the security system and the DL engine (JSON)
2. Determining the statistical significance of the base lines and anomalies.
3. Feeding the engine with mock data.
4. Installing data streams monitoring system.
5. Creating a testing dataset for testing in the upcoming stage.
6. Creating a training dataset for training the DL engine.

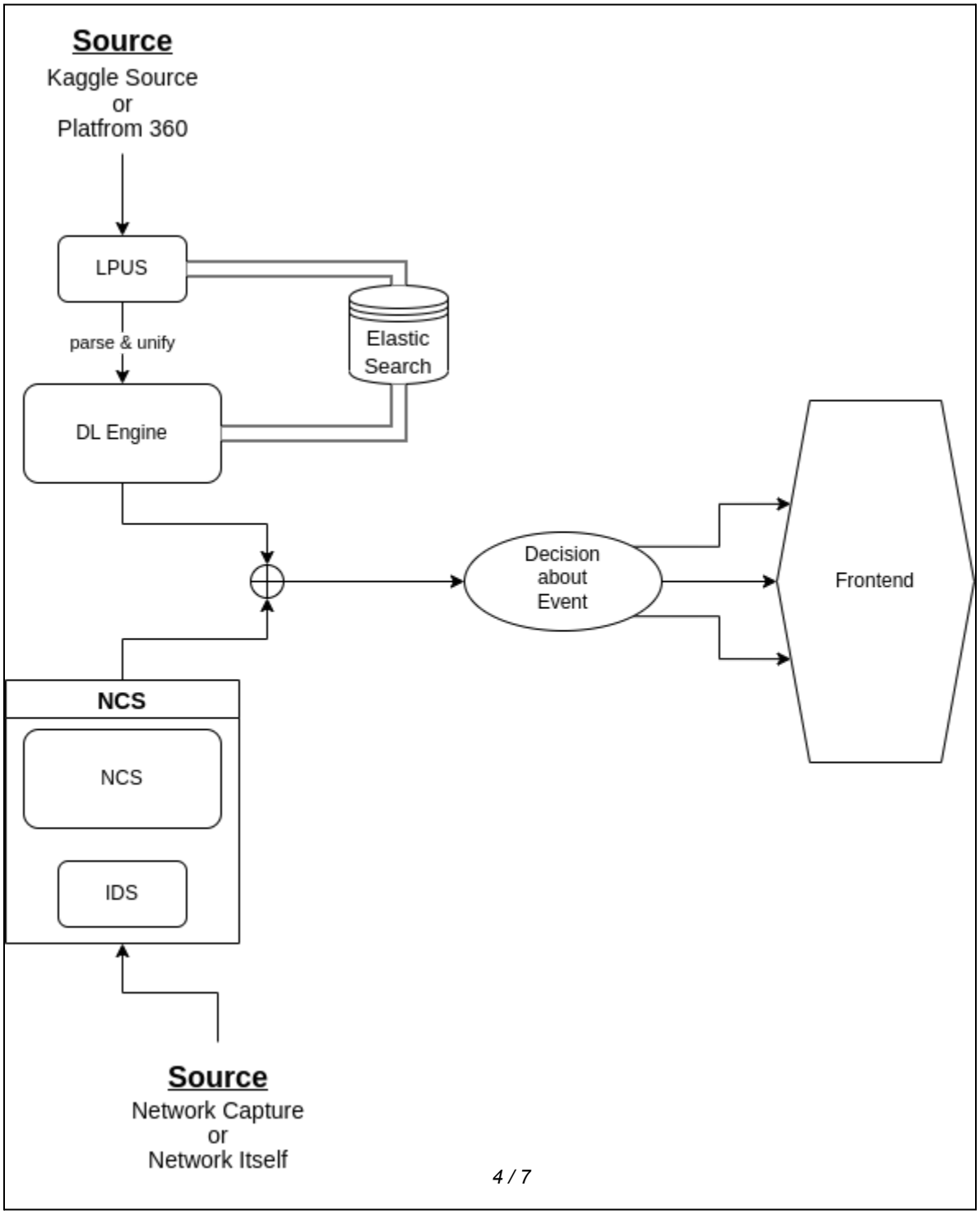
> Related MFs: MF-5, MF-6, MF-9, MF-10, MF-11, MF-12

WP7 - Local testing of DL Engine Attack Detection system with IOT device actions

1. Setting up the testing environment.
2. Testing the overall system with the dataset which is obtained from previous WPs.
3. Check response of security appliances according to signals which are given by the DL system that analyzes behaviors of IOT devices.
4. Detects inaccurate or erroneous responses of security appliances.

> Related MFs: MF-5, MF-6, MF-9, MF-10, MF-11

Overall Systems Architecture



DIONA is a system that encloses three main components: LPUS, DL Engine and Network Control System (**NCS**). Between the LPUS and DL Engine components, there exists an ElasticSearch component which will act as a database and a feedback loop between the DL Engine and the logs generated throughout the lifetime of the project.

LPUS is one of the main components of the DIONA.

- It is responsible for parsing and unifying generated log data for the DL Engine.
- It will preprocess the data generated from the IoT Swarm and pass the processed data to the DL Engine through ElasticSearch.
- This preprocessing period will also be applied to the datasets gathered from services, such as Kaggle, online.

DL Engine is the most important component of the DIONA.

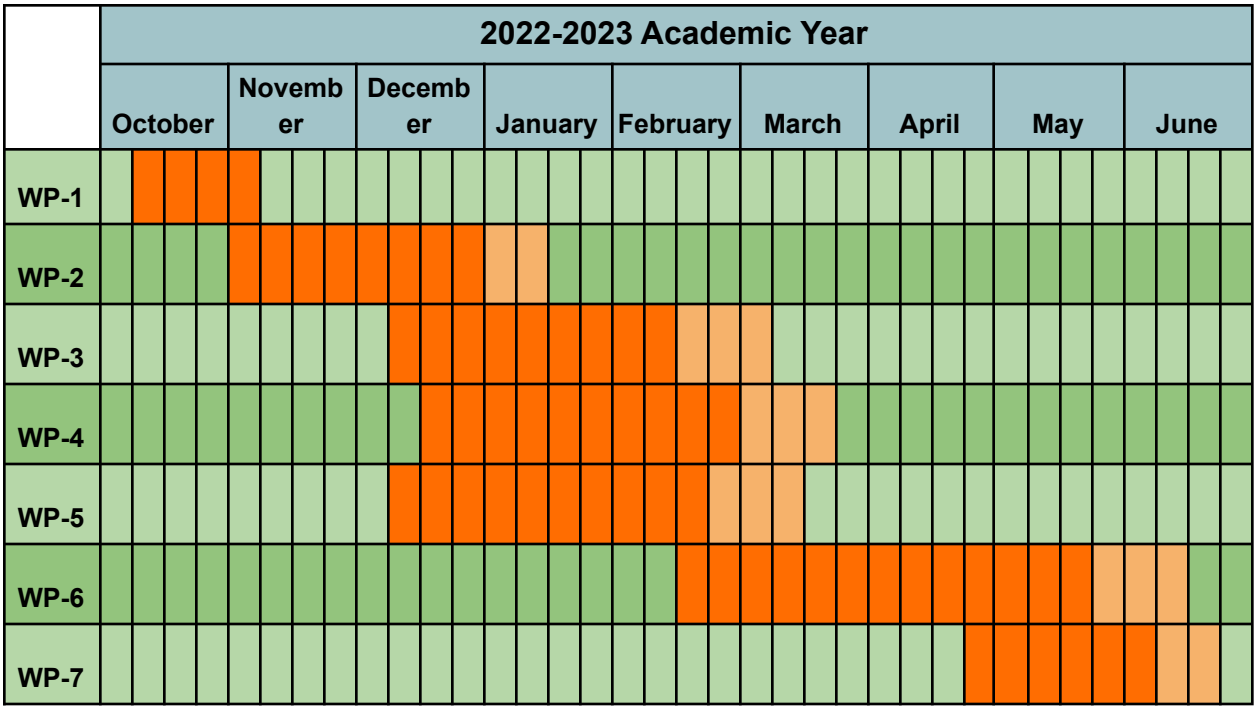
- It will analyze the behavioral data gathered from the IoT Swarm and generate a normal for each IoT device.
- Using these normals and behavioral data, it will detect if there is an anomaly or not.
- By anomaly detection, it will decide on an action using the output generated from the NCS.
- There will be a feedback loop between the LPUS and DL Engine through the ElasticSearch to DL Engine to train itself continuously by using previous log data.

Network Control System (NCS) is the third component of the DIONA.

- Network Control System will detect network intrusions. If there are anomalies detected in the DL Engine, then NCS will look into the logs provided and network traffic to see if there is a network intrusion.
- The Rule-Based Intrusion Detection System (IDS) will be provided with the network traffic and using this traffic it will detect any kind of intrusion based on the predefined rules.

Front-End is the system that is provided for users to analyze and monitor the lifetime of the DIONA. The flow of the information throughout the system and decisions made based on the flow will be provided to the front-end to be watched and analyzed by users. Moreover, users will be able to watch the status from the DL Engine and NCS independently from this user interface.

Timeline


2022-2023 Academic Year
October
November
December
January
February
March
April
May
June
WP-1
WP-2
WP-3
WP-4
WP-5
WP-6
WP-7

Risk Assessment

Risk #	Description	Possible Solution(s)
1	Poorly pre-processed and obtained data may ruin the performance of the overall system, since DIONA depends on the DL Engine as it is the main computing unit.	Researching better feature engineering methods and gathering accurate usage history data.
2	Network traffic may be misleading to detect if there is an intrusion.	Using the anomaly detection service, DL Engine, with the Network Control System will decrease the errors in the intrusion detection process.
3	Lack of configuration and usage information about the IoT devices may result in a longer training period.	Carefully prepared configuration for the specific IoT devices, researching and utilizing the general usage history of such devices.
4	Network intrusion testing may be misleading since sniffing network traffic requires a time interval making the system time dependent. So, it cannot be decided at a real time that there is an intrusion.	The testing environment can be based on a simulation in which a user will be attacking and disturbing the functionality of the system in a pre-calculated time period. After that, seeing if DIONA succeeds in detecting the attack.