Developing a Cyber-security Threat Detection and Alert Systems

Saber Mhiri

***Abstract*—While the industrial infrastructure is increasingly getting connected through changing machines or upgrading them with IoT capacities, the danger of cyber-attacks in their network is increasing too.**
While the industrial network is being upgraded to support the industry move to industry 4.0, the cybercriminals are analyzing and using the vulnerabilities of this new envirement [3] [4] [5] [6] [7].
This situation would obviously concern the network and security responsible of these industrial businesses.
These security responsible are in dire need for simple tools for monitoring and threat detection.
To this end, and based on our previous work [1], in this paper we developed and tested an alert system dedicated to threat detection in an industrial network.

***Index Terms*—OT network, industry 4.0, monitorization, threat detection, threat alerting.**

I. Introduction

With the increase in interconnections between the digital and physical world in industry, the effect of a cyber attack can have severe impact on people's life [2].
Having an alert system that would detect abnormal behavior is a must to survive the industry 4.0 dangers.
To this end, in this work we will start by presenting our alert system based on our previously developed OT_Integrator in the first section. Then, we will evaluate our system with real industrial environment in the second section.

II. Proposed approach

A. Objective

In this work we will focus on defining the different alerts that need to be generated based on the results provided by our OT_Integrator. We will also test our solution on a realistic industrial environment. The alerts will be based on the threat types defined and predicted using our OT_Integrator.
We identify 4 main threats that our system will be identifying and alerting it:

- **New device detection:**
  Using our OT_Analyser with the OT_Integrator we detect any new element that appear in the network. It also detect any change in the protocols used in the network. In the case of a new device appearing in the network, our system identify such threat and launch an alert in order to identify if this new device is authorized.
- **Communication map change:**
  Using our OT_Analyser with the OT_Integrator we detect any change in communication between network devices as defined in [1], such threat would be identified and an alert will be launched in order to identify if the change in communication is due to authorized activities or it is a malicious script running on that PC.
- **Network communication change:**
  Using our OT_Analyser with the OT_Integrator we detect whenever a device start sending or receiving an abnormal amount of traffic as defined in [1]. Our system should be able to identify such threat and launch an alert in order to identify the possibility of a denial of service attack or an unauthorized access...
- **PLC s7 state change:**
  Using our OT_Analyser with the OT_Integrator we learn the state of the PLCs in the network and detect any change in the PLC states that isn't usual as described in [1]. Our system should be able to identify such threats and launch an alert in order to rapidly deal with such threat.

B. Approach description

In order to collect the data in the network, we will be using the OT_Integrator script described in [1]. Then we will apply our OT_Analyser in order to detect the changes in the data according to the previously defined threats and generate alerts accordingly.
The OT_Analyser create a pattern of the collected data described in [1] and analyse the changes in these pattern.

C. Creating Patterns

In order to create the pattern of behavior of the network, we calculate these 3 indicators:

- **Connection pattern** each 2 hours, our OT_Analyser collect all the connection patterns collected with the OT_Integrator [1] and create a resulting table that contain all the communications combinations that appeared during the 2 hours period. The OT_Analyser then save this new table for future comparison.
- **Communication pattern** each 2 hours, our OT_Analyser collect all the communication patterns collected with the OT_Integrator [1] and create a resulting table that contain the average of all the communications that appeared during the 2 hours period. The OT_Analyser then save this new table for future comparison.
- **PLC patterns** each 2 hours, our OT_Analyser collect all the PLC patterns collected with the OT_Integrator [1] and create a resulting table that contain all the PLC combinations that appeared during the 2 hours period. The OT_Analyser then save this new table for future comparison.

D. Scripts and tools

First we have the OT_Integrator that collect the logs, monitor, create and save in a DB the matrix of communication and connection. It also collects and save the PLC state in terms of active and inactive Inputs and Outputs as described in [1]. Second, we also have the script OT_Analyzer that will oversee analyzing the collected data with the OT_Integrator. The OT_Analyzer extracts the saved data and works on detecting threats.
After learning the normal communications in the network through analyzing the matrices of communication and connection, the OT_Analyzer will be able to detect any change in the network in terms of active terminals, the communication protocol type and the flow size in the network. The OT_Analyzer will also analyze the changes in the PLCs and learn the routines the PLCs oversee, then it can detect changes in these PLCs routines.
Based on the results of the OT_Analyzer, the alert modules is a script that will be in charge of physically displaying the alerts to the users.
Finally, in order to test our monitoring system, we developed an attack script that can perform unauthorized direct changes into the state of the PLC outputs.

III. Evaluation and results

A. Testing environment

In order to test our algorithm, we prepared the scenario shown in figure 1.

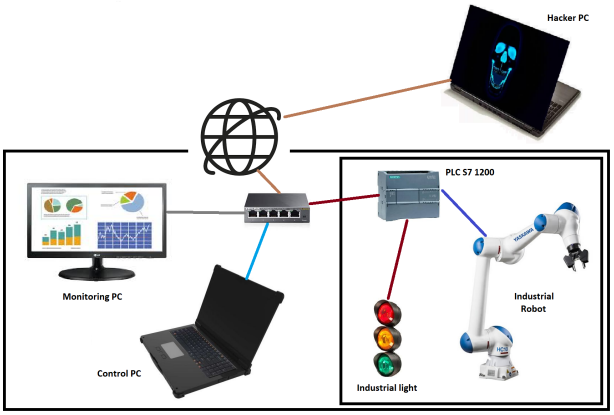

Fig. 1. Testing environment

We created a small industrial network cell that contains, an industrial robot controlled by a S7 1200 PLC; the PLC also control an industrial light that shows the state of the robot to the factory workers. In this case we assigned the red light to indicate that the robot is stopped, the green light to indicate that the robot is working correctly, and the orange light alerts the factory workers that the PLC was hacked.
Outside the industrial layer of the factory, we have the engineer's area where we have the control PC in charge of updating and controlling the PLC state. In this area, our algorithm for monitoring the network using the OT_Integrator and OT_Analyzer scripts are running. The monitoring PC is monitoring the network traffic through the span port of the switch in the network. Through this span port connection (in gray), the monitoring system will receive all the traffic going through the network and will be able to analyze it in order to detect the previously mentioned alerts.
In case of an abnormal behavior detected, our monitoring system will alert the engineering area via the monitoring PC screen and will try to alert the workers in the industrial layer through igniting the orange light.
To test our monitoring system, we developed a script capable of altering the state of the PLC outputs. This script will be implemented in the PC outside the industrial area but connected to it through internet (hacker PC).

B. Evaluation scenario

First step: monitoring the network to learn the normal behavior.
We start by running the OT_Integrator in order to collect the network traffic and save it in the DB. Then we use the OT_Analyzer algorithm to extract the different states the PLC goes through. We assume that the first hour of our test is dedicated for this step and all the network systems are working correctly.
Second Step: testing the alerts module.
After running our OT_Integrator for 1 hour, we managed to create a database containing the different information needed about the network in terms of connection and PLC states and the logs.
We ran our OT_Analyzer algorithm for 30 minutes, while randomly attacking the PLC in the network with the attacking script previously mentioned. We calculated success percentage of our OT_Analyzer script and the average time needed for the script to detect and alert about an attacks.

C. Results

We ran the same simulation 5 times while calculating the P_{detect} and T_{detect} values.

$$T_{detect} = \frac{\sum \text{Time needed to alerts}}{\sum \text{Attacks}}$$

$$T_{detect} = \frac{\sum \text{Rightly detected alerts}}{\sum \text{Attacks}}$$

We can see that when running the experience 5 times, we have an average of 87% success in identifying the threads in the

	P_{detect}	T_{detect}
Total	87%	6 s
Max	100%	10s
Min	62%	4s

TABLE I
THE RESULTS

network. This threat identification was achieved in an average time of 6s. On the other hand the time needed for our script to identify the threats arrived to 10s as a maximum time needed while the least time needed was 4s.

IV. Conclusion

In this work, we have identified some needs for the industrial network protection and developed a solution that can analyze the network and lunch an alert when identifying a threat in the network. The proposed solution where tested and proven success as the results in shown in chapter 3.

References

[1] Saber Mhiri, Network Traffic Analyzing and Modeling for Cybersecurity Threat Detection using AI.
[2] https://www.symantec.com/blogs/feature-stories/cyber-security-predictions-2019-and-beyond
[3] https://e-rse.net/hydrolienne-sabella-d10-energie-marine-22260/gs.kmhj6i
[4] https://globbsecurity.com/ataque-ransomware-metro-san-francisco-40189/
[5] https://www.healthcareitnews.com/slideshow/ransomware-see-hospitals-hit-2016?page=1
[6] https://www.cnet.com/news/wannacry-wannacrypt-uiwix-ransomware-everything-you-need-to-know/
[7] http://www3.weforum.org/docs/WEF_Global_Risks_Report_2019.pdf