Combining exposure indicators and predictive analytics for threats detection in real industrial IoT sensor networks

M. A. Brignoli
Vitrociset – a Leonardo Company

S. Mazzaro
Vitrociset – a Leonardo Company

G. Fortunato
Vitrociset – a Leonardo Company

A. Corà
Vitrociset – a Leonardo Company

W. Matta
Vitrociset – a Leonardo Company

S. P. Romano
University of Naples Federico II

B. Ruggiero
University of Naples Federico II

V. Coscia
University of Naples Federico II

***Abstract*— we present a framework able to combine exposure indicators and predictive analytics using AI-tools and big data architectures for threats detection inside a real industrial IoT sensors network. The described framework, able to fill the gaps between these two worlds, provides mechanisms to internally assess and evaluate products, services and share results without disclosing any sensitive and private information. We analyze the actual state of the art and a possible future research on top of a real case scenario implemented into a technological platform being developed under the H2020 ECHO project, for sharing and evaluating cybersecurity relevant informations, increasing trust and transparency among different stakeholders.**

Keywords —indicators, analytics, big data, threats, iot, ai, cyber security, cyber resilience

I. INTRODUCTION

For the cybersecurity management and implementation, many frameworks and standards have been implemented at sectoral, national, international and global levels. These frameworks and standards propose guidelines that can be adopted, voluntarily and according to needs, by owners and operators of critical infrastructures. Their goal is to allow the identification, evaluation and management of cyber risks. Most security professionals mainly use qualitative and subjective approaches, depending on opinions, insights, type and complexity of organization and ICT infrastructure. But, in order to see how much an infrastructure is "compliant" with standards and best practices, it remains a strong need for independent quantitative measures of network security. Having metrics on IoT sensor networks provides quantitative indicators and enhance them to use predictive analytics, allowing more knowledge and control on the security of the network and on the connected devices. According to the priority areas [18] identified by the European Cyber Security Organization (ECSO), we developed a framework able to automate threats exposure evaluation during the entire life cycle of a product/service, improving preventive capabilities by integrating AI-based tools for a continuous evaluation of security functionalities and of the impact of updates, real-time assessment and patching. One of the expected benefit/impact achieved is the increasing of end-users trust by providing a harmonized vision of cybersecurity risks and traceability of threats evaluation and assessment along the supply chain. The framework supports every organization (SMEs included) in terms of awareness-raising and relevance of "security by design". In practice raises the bar of the security baseline by stimulating competition and better services for the market with less vulnerable products and services deployed. The main idea of this framework is to collect all the opportunities that a hypothetical attacker might exploit, seeing all the possible ways to gain access to the target network. We will show how to effectively prevent attacks by reducing both the attack surface and the attack susceptibility. A metric, collected over time, makes it possible to run comparisons on historical data to properly interpret and control the behavior of a process. At the same time, by leveraging such data within an unsupervised classification algorithm (such as K-Means), we can predict threats with a higher level of confidence. The computation and therefore the prediction allowed by such a metric, helps to define the adequate countermeasures.

II. STATE OF THE ART

In literature there are numerous metrics recently proposed for the measurement of information and network security. The study was done by Weintraub, and Cohoe [1] aims to reduce the uncertainty often linked to risk assessment by proposing objective, quantitative and real-time information that compromises the system's availability. This study has two limitations: the first is the feasibility of graph management that describes the real-time status of each component and its interrelations with other components, including all the safety features. The second limitation refers to the various possible connections between two nodes. In fact, there might be types of connection that do not transfer the attacker to other connected nodes. This problem is a limitation of this model and is also a research problem. The work of Simon Enoch Yusuf's research group [2] proposes a systematic classification of existing security metrics, based on information on the network reachability. The mentioned work classifies security metrics as either host-based or network-based. Host-based metrics are categorized as either "unlikely" or "with probability" metrics, while network-based metrics are classified as either "path-based" or "not based on the path". Indeed, there is a rich literature about graph-based metrics and methodologies [3], [4], [5], [6], [7]. Yusuf's work also presents and describes an approach for developing composite security metrics and calculations using a hierarchical Attack Representation Model via a sample network. This metric does not consider dynamic security metrics. Another method for quantifying the security level of

978-1-7281-4892-2/20/$31.00 ©2020 IEEE

IT networks is that devised by Rashid Munir and others [20]. By electronically scanning the network, using the vulnerability scanning tool, the level of vulnerability in each node is identified and classified according to the Common Standards of the Vulnerability Scoring System (critical, severe and moderate). Subsequently, the probabilistic arguments are then applied to calculate a level of the overall security risk for subnets. Unfortunately, this methodology lacks scalability and automaticity of the solution, fundamental for having reliable real-time measurements, as well as a method able to inject generated traffic on the network. The Center for Internet Security (CIS) also found itself facing the problem of the lack of widely accepted and unambiguous metrics for decision support in the network security field [21]. The CIS work provides a set of standard metrics and data definitions that can be used between organizations to collect and analyze data on performances and results of security processes. Again, this is a simple descriptive solution. Regarding compliances, there are still numerous proposals. The framework uses a risk-based approach to manage cybersecurity risks and is composed by three parts: a core, the implementation tier, and the profiles [22]. It is useful to underline that being compliant does not mean taking the safety. As regarding the use of Big Data analytics and artificial intelligence technologies, Retting et al.[15] describes and empirically evaluates an online anomaly detection pipeline based on Kafka queues and Spark Streaming in order to detect anomalies in Mobile Network using Relative Entropy and Pearson correlation. Hsieh et al [16] proposed DDoS detection based on Neural Networks implemented using Apache Spark clusters. They proposed a system architecture composed by different layers, starting from a packet collector and storing them in a Hadoop HDFS file system in pcap format. These packets are subsequently converted into text files and features are extracted according to the same source and destination IPs, ready to be processed by the neural network. To evaluate their solution, they use DARPA LLDOS datasets like attack traffic, and generate normal traffic using an external application. Mylavarapu et al. [17] proposed a real-time hybrid intrusion detection system using Apache Storm, to run two neural networks: CC4 for anomaly-based detection and a Multi Layer Perceptron for misuse-based detection. The final output of the two Neural Network is handled by a Post-Process unit that gives the classification in the final stage.

III. FRAMEWORK DESCRIPTION

The environment of our framework is based on two mixed-devices test networks which are generally used as test environments for research and development goals, belonging to an Italian company. The entire company network is segmented into sub-networks to serve five sites, on which we distributed intrusion detection probes based on an Open Source IDS (Fig. 1): a Workstation Network, called Network 1, and a Server Network, called Network 2, both hosting heterogeneous IoT devices.

The dataset used was obtained from test networks described below through February 18th and February 20th 2019 for the first run and through February 25th and February 27th 2019 for the second run. These time windows cover workdays (Monday - Wednesday) and work hours (8:00 – 18:30). The data alerts were generated from the probes with these percentages:

- Downadup/Conficker Worm reporting (75%) [9]
- Generic Suspicious Post to Dotted Quad with Fake Browser (5%) [10]
- NETBIOS Stack Overflow Inbound (9%) [11]
- Possible Bad Tunnel for AutoProxy request (3%) [12]
- Non-compliant DNS traffic on DNS Port reserved bit set (5%) [13]
- Potential Scan due to unusual Port 445 and 139 traffic (0,4%) [14]
- Cleartext passwords (0,2%)
- Other (2,4%)

The alerts are spread in the following priorities (starting from the dangerous one):

- Critical (90%)
- Severe (6%)
- High (4%)

It is also important to say that the potential attacks able to steal data from the private network (e.g. Conficker malware) were blocked by Company perimetric Firewall.

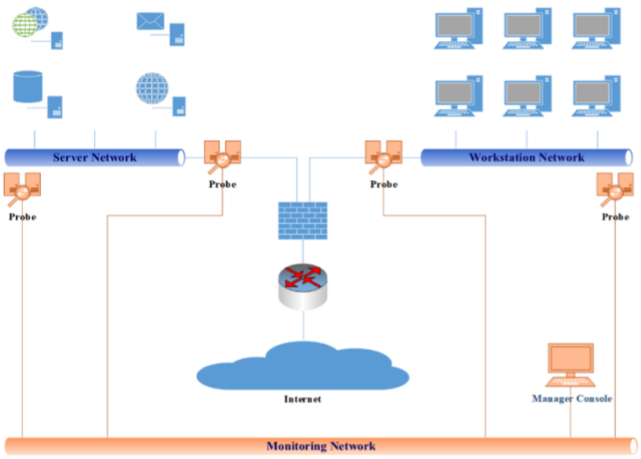

Fig. 1. Test Environment

We divided our exposure indicators into two groups: Attack Surface and Network Susceptibility. The first group includes all the metrics that measure the physical and logical surface that could be subject to a generic attack.

In details, these metrics are:

- Detected Active Hosts (DAH): is a metric that indicates the percentage of systems in the monitored network that was actively detected during the measurement period, and it says how wide it is the attackable surface at the network level;

- Host to Host Interaction (HTHI): is a metric that indicates the established interactions seen during the measurement period and it says how wide is the network level attackable surface related to the kind of interactions, considering that an attacker can easily use

an already established interaction to launch an attack inside the network;

- Services Percentage (SP): is a metric that indicates the exploitable ports to launch an attack, and indicates how big is the transport level attackable surface. It is a combination of the Server Services (SS) and Client Services (CS) metrics.

The second group is characterized by all the metrics that provide a weakness indication of a part or of the whole Attack Surface. Thus, the second group provides a necessary but not sufficient condition for an attack to occur, so we put our focus on attack surfaces.

In details, these metrics are:

- Threat level (TL): is a metric that indicates the common hazard of interactions that generate an IDS alert, considering established interactions more dangerous than new interactions, that in turn are considered riskier than closed interactions. Every IDS alert has a priority from 1 (critical) to 5 (low), so TL1 will have a more significant weight than TL5;

- Severity Average (SA): is a metric that indicates the average severity of the alerts detected, and it helps evaluate the threat exposure;

- Alerts Number (AN): is a metric that indicates how many alerts have been detected;

  Alerted Hosts Percentage (AHP): is a metric that describes the percentage of hosts in the managed network that has been involved in alerts during the measurement process. It helps to evaluate how diverse is the application layer of the destination infrastructure.

Here are the main features of the traffic collected in the network to compute the metrics described below:

- Source IP address
- Destination IP address
- Source port
- Destination port
- Timestamp
- Flow state
- Alert severity

To define a real traffic pattern, we used two of the six days of real NetFlow logs provided by the described testbed, considered as normal traffic, i.e. flows that didn't generate IDS alarms. The dataset has been split into the following percentages:

- 70% for training the framework algorithms
- 15% for validation and thresholds setting
- 15% reserved for testing purpose

Every NetFlow record is composed by the following fields:

Table I. Dataset structure

Field	Description
flow-id	An identifier of the flow
host	The identifier of the capturing probe
ip dst	The destination IP address of the first packet in the flow
ip src	Source IP address of the first packet in the flow
port dst	Destination port number of the first packet in the flow
port src	Source port number of the first packet in the flow
proto	Transport-level protocol
bytes	Total number of flow bytes
flow bytes toclient	Incoming number of bytes associated with the IP flow
flow bytes toserver	Outgoing number of bytes associated with the IP flow
flow pkts toclient	Incoming number of packets associated with the IP flow
flow pkts toserver	Outgoing number of packets associated with the IP flow
flow reason	IP flow interruption reason
flow start	IP flow creation time
tcp flags	Cumulative of all current flow TCP flags in both directions
tcp flags tc	Cumulative of all current flow TCP flags towards the client
tcp flags ts	Cumulative of all current flow TCP flags towards the server
timestamp end	The timestamp of the last packet of the flow

We realized a framework architecture divided into six distinct levels, starting from data acquisition up to data storage. The framework levels are detailed below:

• **Data acquisition**, packets coming from the network that are captured by sensors.

• **Data collection**, traffic captured by sensors, grouped, stored and ready to be processed by the next level.

• **Data preprocessing**, data that are aggregated to obtain "biflow" streams on which exposure indicators are calculated.

• **Features Extraction**, "biflow" data which are aggregated to obtain the desired set of features and that will be used by the threat detection algorithms.

• **Threats Detector**, it performs threat detection on the captured data traffic using K-Means and a classical Flow-Based Detector.

• **Data Storage**, it stores the results of the detection analysis performed and makes them available to the network administrator.

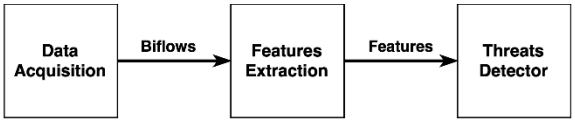

Fig. 2. Data flow diagram

The Features Extractor Module provides the following extracted features to the algorithm:

Table II. Extracted features for K-Means algorithm

Feature	Description
nPkts	Total number of packets
nBytes	Total Bytes
nSrc	Number of different sources
nDstP	Number of different destination ports

As aggregation keys we used the following features:

- Destination IP
- Destination IP, Protocol

To validate the results obtained by K-Means clustering, we tested a function proposed by Myung-Sup Kim et al. [8] to detect Scan based on NetFlow fields and some constant values. The NetFlow is obtained by grouping packets according to the quintuple:

(src_ip, dst_ip, src_port, dst_port, protocol) (1)

The Features Extractor Module accepts as input the NetFlow and aggregates each biflow according to Destination IP. After this aggregation, it processes the features used by the Detector Function.

Table III. Extracted features for Flow-based detection

Feature	Description
nFlows	The number of connections for a single IP destination.
nSrcIP	The number of distinct Source address toward a single IP destination.
nDstPort	The number of different Destination Port toward a single IP destination.
sumFlowSize	Sum of total Bytes toward a single IP destination.
avgFlowSize	Average of total Bytes toward a single IP destination.
devFlowSize	Standard Deviation of total Bytes toward a single IP destination.
sumNumPkts	Sum of the total number of packets in a forward direction toward a single IP destination.
avgNumPkts	Average of the total number of packets in a forward direction toward a single IP destination.
devNumPkts	Standard Deviation of the total number of packets in a forward direction toward a single IP destination.

These features are the input for the Threat Detector Module, which computes an **Anomaly Score** defined as:

$$f_{Scan} = v_{nFlows} * \alpha_{nFlows} + v_{flowSize} * \alpha_{flowSize} + v_{nPkts} * \alpha_{nPkts} + v_{nSrcIP} * \alpha_{nSrcIP} + v_{nDstPort} * \alpha_{nDstPort} \quad (2)$$

A weight is associated with each feature, which can be considered as the main marker in the detection of that particular class of attack, normalized by a specific threshold value associated with the feature, and in particular we have that:

- $vnFlows = nFlows/TnFlow$ (3)
- $vflowSize = TavgFlowSize/avgFlowSize$ (4)
- $vnPkts = TavgNumPkts/avgNumPkts$ (5)
- $vnSrcIP = TnSrcsIP/nSrcIP$ (6)
- $vnDstPort = nDstPort/TnDstPort$ (7)

IV. RESULTS

In Fig. 3 we report the trend of the Attack Surface metrics detected by active hosts percentage, open server and client services percentage and established interactions percentage. On the x-axis of the graphs is present the number of the tests in a range (1.. 30): each interval of five tests refers to a test day. On the y-axis are present the percentage values of the relative metrics. Starting from the graphs is possible to understand that the described trends are not constant. They follow the trends of network traffic during a day, repeating, in a similar way, the other time frames in which the tests were performed.

DAH	SS	CS	HTHI	AHP	AN	TL1	TL2	TL3	TL4	TL5	SA
69.55	48.74	34.72	45.00	1.32	68.07	1.10	0.38	1.44	0	0	4.62

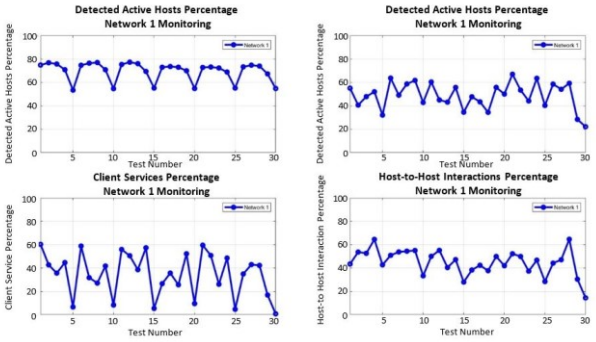

Fig. 3. Network 1 Mean Results & Attack Surface Metrics

In Fig. 4 it is clear that the four metrics of Network 2, relative to the Attack Surface, take low values. Furthermore, all the metrics belonging to this group assume a behavior over time that has an almost constant profile. From this, it can be deduced that the Server Network 2 attack surface profile is relatively limited and with almost a consistent behavior.

DAH	SS	CS	HTHI	AHP	AN	TL1	TL2	TL3	TL4	TL5	SA
8.74	4.36	4.84	7.83	0.26	0.93	0.10	0	1.10	0	0	1.73

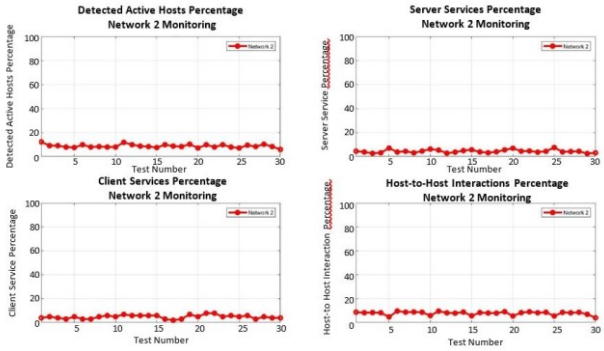

Fig. 4. Network 2 Mean Results & Attack Surface Metrics

In Fig. 5, the threat detector module can be appreciated in action, running on the same dataset as Network 1 and Network 2 and using K-Means for threats classification.

In particular, after the training phase, we ran random port scans and DoS/DDoS attacks from different nodes of the network.

The figure shows the results achieved using different aggregation keys for each biflow, i.e. (DestinationIP) and (DestinationIP, Protocol). In particular, choosing K=2 as a number of clusters, and considering (DestinationIP, Protocol) as aggregation key, we were unable to achieve a separation between anomalous and benign traffic, unlike the case in which the only (DestinationIP) is used.

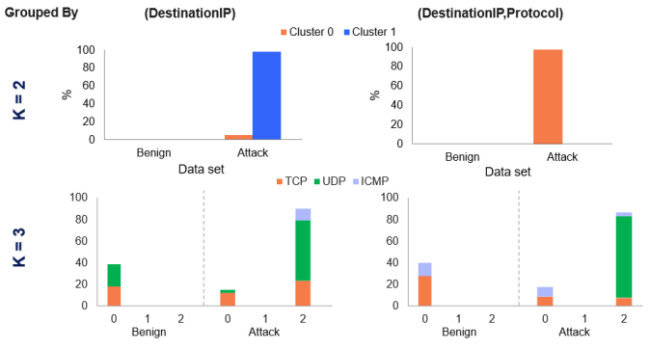

Fig. 5. K-Means in action from the threat detector module

Fig. 6 shows the results running the flow-based detection function in comparison with the normal traffic dataset and the attack dataset:

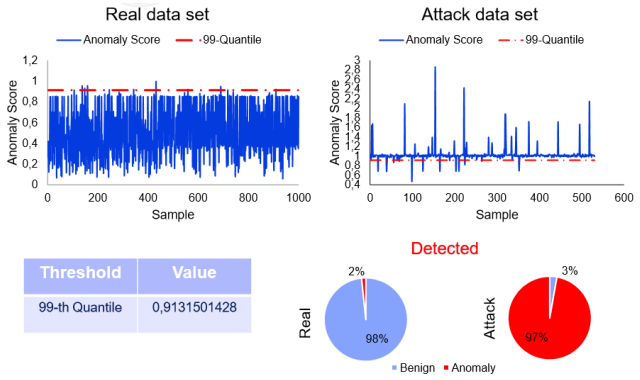

Fig. 6. Results for Flow-Based Detector

In particular, the threshold value was chosen by applying the flow-based detection function on real traffic dataset. The graph related to the attack dataset shows how the traffic patterns get a higher value of anomaly score than the threshold foreseen by the benign traffic pattern

A statistical view of the anomaly Score for both datasets is reported in Table IV. Statistical view of the Anomaly Score:

Table IV. Statistical view of the Anomaly Score

	Normal-traffic dataset	Attack dataset
Mean	0.5386485087	1.0421997219
Median	0.4844107364	1.0149445714
Std Dev	0.2257778374	0.3213015025
Min	0.0299296001	0.4426064366
Max	1.7601188363	2.9904500056

In addition to 99-th quantile, many other types of thresholds have been assessed. Normal-traffic dataset thresholds values and related anomaly percentages are reported in Table V:

Table V. Different thresholds in normal traffic dataset

Threshold type	Threshold value	% Anomaly
95-th Quantile	0.8500092094	4.3557 %
99-th Quantile	0.9131501428	0.4133 %
Mean + Std	0.7644263483	25.7330 %
Mean + 3*Std	1.2159820232	0.0055 %
Median + Std	0.7091543706	25.9336 %
Median + 3*Std	1.1602104452	0.0076 %

Table VI shows the attack dataset results:

Table VI. Different thresholds in attack dataset

Threshold type	Threshold value	% Anomaly
95-th Quantile	0.8500092094	97.9843 %
99-th Quantile	0.9131501428	97.2004 %
Mean + Std	0.7644263483	98.4322 %
Mean + 3*Std	1.2159820232	4.1433 %
Median + Std	0.7091543706	98.4322 %
Median + 3*Std	1.1602104452	4.3673 %

V. Application Scenario

Currently, the discussed framework is deployed and act as an EWS-Plugin, providing its capability to the ECHO Early Warning System (EWS). Under the ECHO Project founded by the European Union's Horizon 2020 Research and Innovation Programme [19], it is implementing the EWS to provide secure information sharing, cybersecurity alerts and actionable insights to identify, respond, prevent and mitigate cybersecurity threats to benefits citizens, companies and governments. Several modules compose the EWS-Plugin system, and a logic view is represented in Figure 7. The Service Manager Console (SMC) is providing platform management, common functionalities (IDS rules) and AI-capabilities with functionalities to present relevant aggregate data. Probes are system elements responsible for collecting network data traffic from a LAN segment. Probes analyses traffic and transmit NetFlow protocol data and IDS alerts (from the intrusion detection system) to the Console Manager (CM). CM is the local focal point for all data collection, processing and correlation ideally located in the probes LAN.

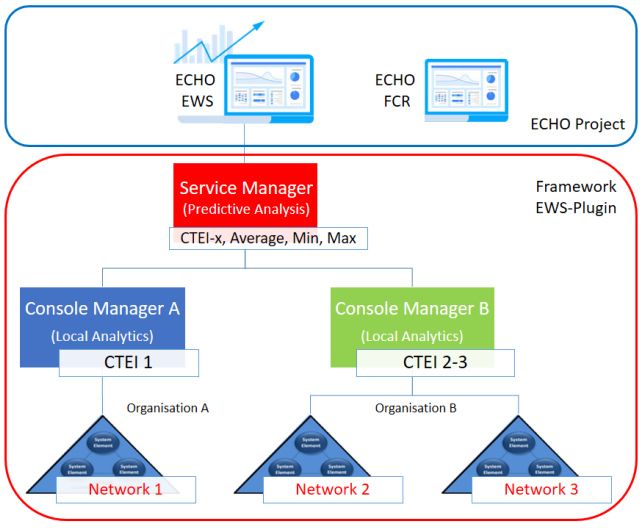

Figure 7 - ECHO EWS-Plugin

Privacy is a big concern inside the ECHO project, and the described framework provides flexibility to minimize any concern about it. LAN traffic is analyzed in the CM, inside the organization security perimeter, and only aggregate indicators like CTEI (Cyber Threat Exposure Indicator) are sent to other external systems, like E-EWS in a secure way over the Internet.

VI. Conclusions and Future Research

We presented and tested a framework to automate threats exposure and evaluation during the entire life cycle of an ICT infrastructure, improving preventive capabilities by integrating AI-based tools for continuous evaluation of security functionalities and the impact of updates, real-time assessment and patching.

In conclusion, we can say that the definition of a single integrated figure describing the security of the network is more efficient than relying on several parameters to be correlated. With a single reference number, an IoT network security operator could instantly understand that something went wrong and he could be able to immediately start analyzing the reasons that let the overall security fall down. Also, we can use the indicator, and its prediction, for two different purposes:

- To continuously monitor a production IoT network exposure and find anomaly behaviors if the indicator is very far from the mean.
- To evaluate the impact of active countermeasures to improve the security of the IoT network, by measuring the difference between the indicator values right before and after their application.

Framework's scalability is also a core aspect of providing security evaluation/testing techniques. Time to market and cost factors are also critical, especially for SMEs. The potentially large number of devices to be evaluated requires the design of cost-effective evaluation procedures. These are all points that need to be properly addressed in order to define any business model to exploit the framework potentials.

The main future target will be the improvement of the performances, some of the possible ways to achieve this goal are:

- Building a system that always keeps the IDS rules engine updated, mainly to reduce the probability that a malicious flow of traffic could be considered normal, invalidating the considered dataset;
- Testing this architecture using more complex algorithms, for example those based on Deep Learning and Reinforcement Learning, to increase the accuracy and the reliability of the obtained results.

VII. References

[1] E. Weintraub and Y. Cohen, "Defining Network Exposure Metrics in Security Risk Scoring Models," International Journal of Advanced Computer Science and Applications, Tel Aviv, Israel, 2018.

[2] S. E. Yusuf, J. B. Hong, M. Ge and D. S. Kim, "Composite Metrics for Network Security Analysis," Journal of Software Networking, pp. 137-160, 2017.

[3] N. Idika and B. Bhargava, "Extending Attack Graph-Based Security Metrics and Aggregating Their Application," IEEE Transactions on Dependable and Secure Computing, vol. 9, no. 1, pp. 75-85, 2012.

[4] J. Pamula, S. Jajodia, P. Ammann and V. Swarup, "A weakest adversary security metrics for network configuration security analysis," in Proceedings of the 2nd ACM workshop on Quality of protection - QoP '06, New York, NY, 2006.

[5] L. P. Swiler and C. Phillips, "A Graph-Based System for NetworkVulnerability Analysis," New York, NY, 1999.

[6] L. Wang, T. Islam, T. Long, A. Singhal and S. Jajodia, "An Attack GraphBased Probabilistic Security Metric," in Data and Applications Security XXII, Springer, Berlin, Heidelberg, 2008, p. 283–296.

[7] L. Wang, A. Singhal and S. Jajodia, "Measuring the Overall Security of Network Configurations Using Attack Graphs," in Data and Applications Security XXI, Berlin, Springer, Berlin, Heidelberg, 2007, pp. 98-112.

[8] Myung-Sup Kim et al. "A Flow-based Method for Abnormal Network Traffic Detection". In: Jan. 2004, pp. 599–612. DOI: 10.1109/NOMS.2004.1317747.

[9] https://www.microsoft.com/en-us/wdsi/threats/malware-encyclopedia-description?Name=Worm:Win32/Conficker.B

[10] https://www.hybrid-analysis.com/sample/f5813aff8e773b54a1fbcc5c7514128a90ee1bf67d25e5c290a76261b8ad87fe?environmentId=100

[11] https://nvd.nist.gov/vuln/detail/CVE-2008-4250

[12] tools.ietf.org/html/draft-ietf-wrec-wpad-01

[13] https://blog.talosintelligence.com/2008/08/checking-multiple-bits-in-flag-field_29.html

[14] https://metaflowsblog.wordpress.com/2017/05/15/wannacry-ransomware-advisory/

[15] Laura Rettig et al. "Online anomaly detection over Big Data streams". In: *2015 IEEE International Conference on Big Data (Big Data)* (2015), pp. 1113–1122.

[16] Chang-Jung Hsieh and Ting-Yuan Chan. "Detection DDoS attacks based on neural-network using Apache Spark". In: *2016 International Conference on Applied System Innovation (ICASI)* (2016), pp. 1–4.

[17] Goutam Mylavarapu, Johnson P. Thomas, and K Kumar T. AshwinKumarT. "Real-Time Hybrid Intrusion Detection System Using Apache Storm". In: *2015 IEEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th International Symposium on Cyberspace Safety and Security, and 2015 IEEE 12th International Conference on Embedded Softwaree and Systems* (2015), pp. 1436–1441.

[18] ECSO – European Cyber Security Organisation STRATEGIC RESEARCH AND INNOVATION AGENDA – June 2017, pp. 8-9 https://www.ecs-org.eu/documents/publications/59e615c9dd8f1.pdf

[19] ECHO - the European network of Cybersecurity centres and competence Hub for innovation and Operations – (Grant Agreement no 830943) https://echonetwork.eu/

[20] R. Munir, J. Pagna Disso, I. Awan and M. Rafiq, "Quantitative Enterprise Network Security Risk Assessment," in UK Performance Engineering Workshop, July 2013.

[21] CIS, Center for Internet Security, CIS Security Metrics v1.1.0, November 2010.

[22] M. P. Barrett, "Framework for Improving Critical Infrastructure Cybersecurity," Cybersecurity Framework, no. Version 1.1, April 16, 2018.

[23] Laura Rettig et al. "Online anomaly detection over Big Data streams". In: *2015 IEEE International Conference on Big Data (Big Data)* (2015), pp. 1113–1122.

[24] Chang-Jung Hsieh and Ting-Yuan Chan. "Detection DDoS attacks based on neural-network using Apache Spark". In: *2016 International Conference on Ap- plied System Innovation (ICASI)* (2016), pp. 1–4.

[25] Goutam Mylavarapu, Johnson P. Thomas, and K Kumar T. AshwinKumarT. "Real-Time Hybrid Intrusion Detection System Using Apache Storm". In: *2015 IEEE 17th International Conference on High Performance Computing and Com- munications, 2015 IEEE 7th International Symposium on Cyberspace Safety and Security, and 2015 IEEE 12th International Conference on Embedded Soft- ware and Systems* (2015), pp. 1436–1441.