http://dx.doi.org/10.14236/ewic/ICS2018.10

On the Edge Realtime Intrusion Prevention System for DoS Attack

Rishabh Das
Department of Electrical and Computer Engineering
The University of Alabama in Huntsville
Huntsville, USA
rd0029@uah.edu

Vineetha Menon, Ph.D.
Department of Computer Science
The University of Alabama in Huntsville
Huntsville, USA
vineetha.menon@uah.edu

Thomas H. Morris, Ph.D.
Department of Electrical and Computer Engineering
The University of Alabama in Huntsville
Huntsville, USA
tommy.morris@uah.edu

Industrial control systems manage critical infrastructures that are immensely diverse and complicated. These highly linked critical infrastructures are made up of networks of industrial control system (ICS) each responsible for controlling critical processes. During its nascent stages the controllers in the ICS were built for robust operation in extreme industrial conditions, but little to no emphasis was placed on safeguarding the system against potential cyberthreats. The industrial networks having legacy controllers are air gapped from the enterprise network hence a centrally deployed NIDS in the same network of the trusted nodes is often used as the last line of defence against intrusions such as malicious activity or policy violation. Most cyber incidents in industrial control systems have witnessed the breach of the air gap and compromised trusted nodes. Hence this paper proposes an on-the-edge Intrusion Prevention System (IPS) that can detect and prevent Denial of Service (DoS) attack on the Programmable Logic Controllers (PLCs) from trusted nodes at real time. A novel attribute of our proposed framework is that it is generic in nature and can be used on any PLC irrespective of the critical infrastructure being controlled by it. A wide range of experimentation has been performed to validate the performance of our proposed IPS.

SCADA, ICS, on-the-edge Intrusion Prevention System, Denial-of-Service attack, Unsupervised Machine Learning.

1. INTRODUCTION

Supervisory Control and Data Acquisition (SCADA) systems and their components (PLCs, sensors, actuators, etc.) are used to monitor and control the highly interconnected subsystems and complex physical processes in critical infrastructures. The physical components of an ICS are highly resilient to the harsh industrial environment. However, cyber security threat was not a major concern during in the structural design and composition of initial ICS components. “Security by obscurity” was the prime approach employed in the industry [2], but the emergence of new technologies (like smart sensors, remote monitoring and control, etc.) compelled a major confluence of the information technology (IT) and operational technology (OT) networks. The PLCs once air-gapped are no longer isolated from the enterprise network. Consequently, exposing ICS to a plethora of cyber threats. According to a report released by IBM, attacks against ICS have increased by 110% in 2016 compared to 2015 [1]. It is of utmost importance to develop a secure framework to safeguard ICS components through early detection of malicious behavior and in turn prevent process control disruption of PLCs.

The industrial controllers are vulnerable to TCP/ IP DoS attacks such as SYN flooding and low-rate DoS (LDoS). Even informational network activities like port scanning on operational network can significantly slow down or halt the system [3]. The Mirai botnet, which was first encountered in September 2016 compromised millions of IoT (Internet of Things) devices and was used to perform directed volumetric DoS attack on a DNS provider named Dyn. This attack managed to disrupt major websites like Spotify, Twitter and PayPal [5]. Similarly, the Bricker bot targeted all vulnerable IoT devices running BusyBox application and rendered them inoperable by performing a permanent denial of service(PDoS) attack. A PDoS attack is lethal as it causes permanent hardware damage and any software patch ensued or even replacement of the entire system software cannot reinstate the system operation [4]. To protect legacy devices from attackers we propose implementation of IPS inside the peripheral control components that directly control critical processes. Proposed framework

would introduce a multi-layer approach to Cybersecurity, because even if a trusted node in the network with direct access to the controller is compromised, the attacker must still go through an additional on-the-edge line of defense to gain access to control system. In this paper we propose a machine learning based IPS that safeguards the legacy controllers against volumetric denial of service attacks.

The main contributions of this paper are as follows: (1) design and implementation description of a novel IPS framework, (2) real-time performance analysis and experimental validation of the proposed IPS. The remaining sections of this paper are organized as follows: related works, physical layout of on-the-edge IPS, internal architecture of IPS, results and conclusion.

2. RELATED WORKS

Centralized intrusion detection systems have been a core part of the SCADA. Normally, the IDS reside on the central server or gateway, at the confluence of the information and operational network. The incoming and outgoing traffic is continually monitored by the IDS. Ideally, any encountered malicious packet or command from an external network (as internet) will be promptly blocked and sieved by the IDS. Since the operational network is often air-gapped from the informational network. All network traffic and commands from the trusted nodes like Human Machine Interfaces (HMIs) and end computers in the operational network are considered legitimate and are overlooked by the firewall and the IDS. But in recent years the world has seen multiple airgap breaches and the compromised trusted nodes. Popular attacks like STUXNET [10] and Shamoon [9] were able to compromise these trusted nodes. Hence an IPS embedded inside the PLC in addition to the centralized IPS, will be an effective cyberattack preemption from trusted nodes. This IPS will reside at the edge of ICS, i.e. at ICS interface between operational network and the physical controller device. In the event of a compromised ICS network, the proposed embedded IPS is intended to act as the last line of defense against cyberattacks.

Although the concept of on-the-edge IPS algorithm is novel, there has been some prior work to investigate the applicability of machine learning algorithms for real-time intrusion detection for critical infrastructures. Adhikari et al. proposed a real-time framework that can be used to detect traditional cyber contingencies and cyber-attacks. The framework had a cascade of three approaches Hoeffding Adaptive Trees (HAT), drift detection method (DDM) and adaptive windowing (ADWIN) and achieved an accuracy of over 94% while detecting 45 different cyber contingencies. The framework uses labelled data for training the classifier [12]. A multitude of systems are being controlled by the industrial controllers and it challenging to obtain good real-time data to train supervised algorithms. The use of supervised algorithms makes the framework domain-specific. The caveats of the analysis by Adhikari et al. using supervised learning include, 1) Segmented analysis can be ineffective if the data acquired is not representative of the entire process. 2) the performance of the framework has not been tested with streaming online data. 3) The framework has not been implemented on computers that are computationally less capable and hence it is not possible to ascertain the performance of the algorithm. Mowla et al. developed a framework using a combination of artificial neural network and decision tree to detect intrusions in medical devices. Although the analysis portrayed optimistic detection rate of 95% the framework was tuned for a specific data set. Hence the scalability of any framework with supervised algorithm depends on the labelled data being used. This is a drawback of any supervised algorithm and the performance of the algorithm may vary with the dataset [19].

In contrast, the proposed work uses unsupervised learning to study the network properties in real-time. This unsupervised learning makes the IPS generic and it can be used on any SCADA system irrespective of the application process being controlled. Since IPS is implemented on-the-edge (Inside the controller) it impedes the attackers capability to remotely tamper with the functionality of IPS. Finally, most Machine Learning (ML) based cyber intrusion detection framework detects cyber-attacks from offline data and are meant to be used for data analytics. The proposed framework detects the DoS attack and blocks the attacker from accessing the PLCs.

3. PHYSICAL LAYOUT OF ON-THE-EDGE INTRUSION PREVENTION SYSTEM

Alves et al. proposed an efficient way of modularizing any generic SCADA system for virtualization of complicated physical process [6]. This modularized architecture helps SCADA researchers to better understand and analyze the modifications that are to be incorporated to a generic SCADA system.

3.1 Modular SCADA Architecture

A SCADA system can be effectively modularized into five major components; physical system, cyber-physical link, PLC along with the I/O the SCADA protocol and the Human Machine Interface (HMI). The physical system constitutes multi domain dynamic systems which include a wide

variety of critical infrastructure (Pumping station, Electrical sub-station, Cryogenic gas pipeline,etc.) being monitored by the industrial controllers. The cyber physical link is the physical connection between the physical system and the I/O modules of the PLC. The PLC is an intelligent industrial computer that constantly monitors the connected sensors and actuates any connected peripheral device autonoumously according to preprogrammed logic or manually according to defined user input through the HMI. The network SCADA protocol is a specialized network protocol used by industrial controllers. This protocol is used by the HMI to query the PLC about the status of the system or to write a response of a user. The status of the physical system is visually illustrated to the user using the HMI as in figure 1.

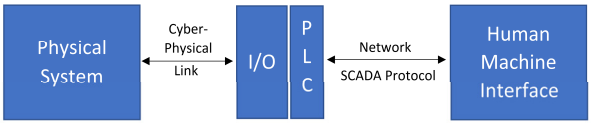

Figure 1: *SCADA components in a generic industrial control system*

3.2 Proposed SCADA Architecture

The proposed IPS can be incorporated to any generic SCADA controller which supports industrial protocol encapsulated under TCP/ IP header. Most commercial vendors do not release the source code of their software module running on the PLCs, as well as some PLC hardware available on market does not support POSIX operating system. Therefore, two distinct network architectures are proposed to overcome this challenge and incorporate the IPS effectively.

3.2.1 Firmware Update

If the software module of the PLC is running on the top a POSIX operating system, a firmware update can be developed to effectively incorporate the IPS inside PLC and no external module (hardware) would be necessary. Most modern PLC software runs on top of a POSIX compatible operating system. Hence this approach might be more cost effective and convenient since there is no modification to the physical layout of the system. Figure 2 demonstrates how the SCADA modules would interact with the modified PLC having the embedded IPS. The firmware update would not affect the operation and interaction of the other modules of the system with the PLC, only the traffic from the external network will be intercepted by the IPS internally and forwarded to the PLC only if the received network packet is deemed safe by the IPS. The response of the PLC would be redirected to the IPS and in turn the IPS would forward the response of the PLC to the HMI.

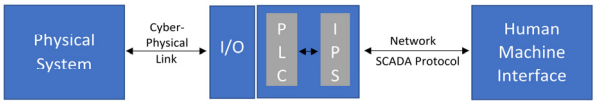

Figure 2: *IPS Embedded inside the PLC*

3.2.2 External module

The IPS can be implemented as an external module connected in front of the PLC. Any inexpensive hardware processor module (like Raspberry Pi [13-14]) capable of running POSIX based operating system can be used to implement the IPS framework. The interaction of the modules of the SCADA system after external IPS is included would be similar in function to the embedded IPS. Figure 3 demonstrates the interaction of different SCADA module after the implementation of the IPS as an external module.

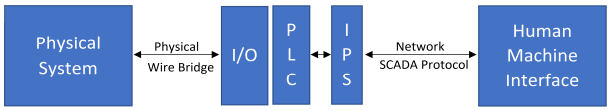

Figure 3: *IPS as an external module*

4. INTERNAL ARCHIETECTURE OF THE INTRUSION PREVENTION SYSTEM

An open source PLC platform (OpenPLC [7]) is used for implementing the IPS. The IPS is added as an extra module to the source code of the OpenPLC and no direct alteration is done. The IPS module works alongside the PLC process and is embedded in the controller at the edge. This modification is possible because the open source code of the OpenPLC is not black boxed from users and researchers. The secure version of the OpenPLC is implemented on an industrial UniPi board. The PLC is configured to use MODBUS TCP/IP as the SCADA protocol. Python 3.6, scikit-learn, pandas and Mathplotlib are installed in the PLC module to make the IPS module work.

During commisioning the data collector module of the IPS collects data related to packet interarrival time and packet processing time for all incoming data packets. While the data is being collected, the received packets are forwarded to the PLC. When the required data has been collected, a dedicated thread preprocesses the data by deleting the tuples containing the outliers using local outlier factor (LOF) algorithm. The processed data is used by the K-Means algorithm to cluster the packets' interarrival time and the packet processing time. After the initial training, the model is deployed and the IPS starts clustering the incomming traffic but the incident respose system of the IPS is not initiated and more data is collected to optimise the clusters centers. After a second iteration of training of cluster centers using K-means, the embedded

IDS is starts monitor the incoming data stream for cyber intrusions and the incident response system of the IPS is activated. The proposed IPS module constantly updates the monitoring model after every 200 samples. The rest of the section describes in detail how the IPS is incorporated in the PLC framework and how the communication channels are modified with the embedded IPS.

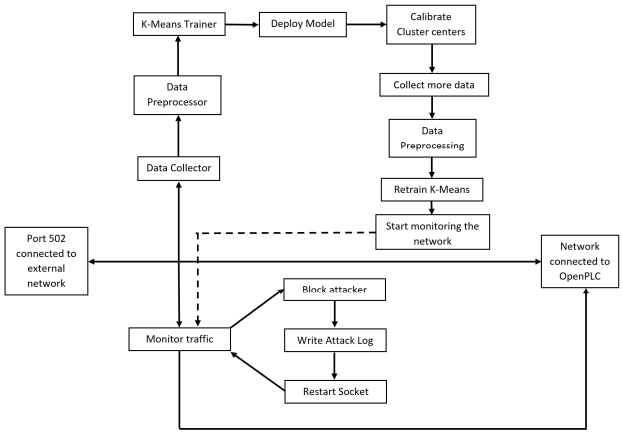

Figure 4: *Internal Flow of control in the IPS*

4.1 Relay Server Module

The IPS hosts a TCP proxy server which acts as an intermediary client while interacting with the PLC and acts as a server while receiving command from the trusted nodes. Hence, this module gives IPS the capability to receive and forward the incoming packets to the PLC after filtering out the anomalous traffic. If multiple HMIs are communicating to the PLC the proxy server accepts multiple client connections from the trusted nodes. For each connection the proxy server initiates a new connection to the PLC.

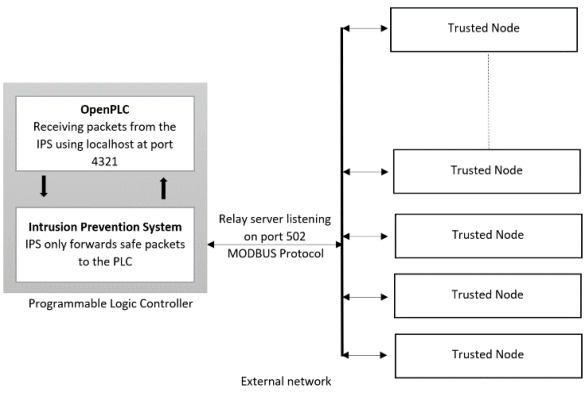

Figure 5: *Network Layout of the implemented IPS*

The relay server module of the PLC encapsulates all other module inside it. All other external connection to or from the IPS is handled by the relay server module.

In a typical MODBUS communication, the PLC listens on port 502 for incoming traffic. OpenPLC allows the users to configure the listening port as a runtime command line argument. This feature is used to make OpenPLC listen on port 4321 instead of the conventional 502 for incoming MODBUS communication. The relay server module of the IPS listens on port 502 for any incoming traffic and forwards any received data packets from the trusted nodes to port 4321 of the OpenPLC through the internal loopback network of the POSIX based operating system. The response of the OpenPLC is communicated back to the IPS through the internal network. Which is further communicated back to the trusted nodes by the relay server module. Figure 5 shows the internal and external network of the PLC. The access to port 4321 is made available through the internal network of the operating system and all traffics from any external network entity is prevented using IP table rules.

4.2 Data Collection

When the PLC is commissioned for controlling a new SCADA system, data tuples comprising packet interarrival time and packet processing time of the network is collected real-time by the data collector module of the IPS. During the data collection process, the received packets are forwarded to port 4321 of the PLC. Once the required data has been collected it is processed by a dedicated thread.

The packet interarrival time is calculated by collecting the associated timestamp of any network packet being received by the relay server. Once a second packet is received from same client the time stamp of the second packet is subtracted from the stored time stamp of the previous packet to find the network latency associated with the concerned client. Finally, the stored timestamp is replaced by the newer one. This attribute is calculated by the data collection module right away once any network packet is received by the relay server module. Processing time of a packet is the time duration spent by the PLC to process a network packet, perform the required operations associated with the network packet and to respond to the relay server with a valid response. To calculate this attribute once a network packet is received, a counter is started by the relay server module and the packet is forwarded to the PLC. When the response for the packet is received from the PLC the value of the counter is stored as the value of the attribute in the data tuple. Hence the packet interarrival time and the processing time is matched for a distinct network packet and is stored as a tuple of data in the dataset being constructed. For every 200 tuples the dataset is sent to the data preprocessor.

4.3 Data Preprocessing

Due to collision of network packets sometimes the network latency recorded is not consistent and the values are much larger than expected. The

scheduling of the process running the IPS is handled internally by the POSIX operating system and sometimes the processing time is longer than the expected whenever the process is scheduled out by the operating system; this increases the processing time of the packets by the PLC. Deterministic clustering algorithms like bisecting K-Means is very sensitive to noise and the value of the cluster centers are distorted in the presence of outliers hence before training the algorithm the outliers are removed from the training dataset [15].

Local outlier factor (LOF) algorithm computes a score reflecting the degree of abnormality of the observations [18]. The local density of the data point with respect to its neighborhood is computed for each point. Typically, the LOF score of a data point is the ratio of the average local density of k-nearest neighbor to its own local density. Since the global and local attributes of the dataset is considered in this algorithm it can perform well even when samples have different underlying densities. The points having lower density is considered as outliers in the given data set and the corresponding tuples are removed. This preprocessed data is forwarded to the machine learning module for training the IPS.

4.4 Machine Learning Module

This module learns the characteristics of the processed data to apprehend the normal behavior of the network in which the PLC is deployed. Once the module is trained, the incoming network traffic to PLC is analyzed real-time and is compared with the prior learnt network behavior by the traffic monitoring module. The network behavior is learnt using unsupervised bisecting K-Means algorithm [15]. The bisecting K-means algorithm creates required number of clusters based on the preprocessed data.

Bisecting K-means clustering is an unsupervised algorithm, which partitions *n* number of samples into K clusters where each sample is assigned to the cluster with the closest mean. The bisecting K-means algorithm provides a hard classification of the samples by providing only definite memberships of each sample to the assigned cluster. Let $\mathbf{X} = \{\mathbf{x}_1, \ldots \mathbf{x}_n\}$ be a set of *n* observations, where $\mathbf{x}_i \in R^d$. K-means aims at partitioning the given data samples into K clusters where K<<n such that the data variance in each cluster is maximized.

Let the set of data points belonging to K clusters be denoted as $\mathbf{S} = \{\mathbf{S}_1, \mathbf{S}_2, \ldots \mathbf{S}_K\}$, where $S_i = U\ \mathbf{x}_j$, for all $\mathbf{x}_j \in$ cluster i and j=1,..n the objective function can be written as:

$$\arg\min_{s} \sum_{i=1}^{k} \sum_{x \in S_i} x - \mu_i^{\,2} = \arg\min_{s} \sum_{i=1}^{k} |S_i| Var\, S_i$$

Where μ_i is the mean of the points in S_i

4.5 Traffic Monitoring Module

The trained IPS model from the machine learning module provides the traffic monitoring system with the cluster centers. These cluster centers are used to compute the Euclidean distance of the cluster centers from the new data points from the incoming traffic to track the changes in the cluster centers. If $\mathbf{x} = (x_1, x_2, x_3, \ldots\ldots, x_n)$ and $\mathbf{y} = (y_1, y_2, y_3, \ldots\ldots, y_n)$ are two points in a n-dimensional Euclidean space then the Euclidean distance **d** from **x** to **y** is represented by the formula:

$$d(x,y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \ldots + (x_n - y_n)^2}$$

$$= \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

The distance from the two-dimensional cluster centers (n=2, because only two features are used in the framework) is measured from the incoming network packets individually. A normal cluster region is predefined in the module. Whenever a new distance is found to lie outside the normal cluster region, the packet is deemed anomalous and is not forwarded to the PLC.

In the event of a DOS attack, the same client will start sending packets at a much higher rate than the normal scenario. The packet interarrival time from the client is noted to drastically decrease and is thereby flagged as an attack by the traffic monitoring module. The IP address of the suspected attacker is forwarded to the incident response module for further action.

4.6 Incident Response System

This module protects the PLC from the detected attacker. Once an attacker is earmarked, a custom IP table rule is generated and executed. This rule drops all incoming traffic from the attacker's IP and stops the flood of network packets from the attacker's computer. After executing the IP table rule an incident report is generated about the attempted DOS attack. To discard the network packets used for flooding the network adapter the network socket is restarted by the module. This causes a brief interruption in the communication but flushes out all the network packets sent by the attacker and refreshes the socket. During the restart the OpenPLC program is unaffected and keeps monitoring the SCADA processes. Finally, the IPS starts monitoring the network for cyberattacks again.

5. RESULTS

The DOS attack hinders the communication between two nodes in a network [16]. In case of

volumetric DoS, the attacker disrupts the communication between nodes by flooding the network by a surge of network packets. For legacy controllers this attack can be detrimental, since these huge volumes of packets can often overwhelm the controller and stall the critical physical processes. An opensource tool called Low Orbit Ion Cannon (LOIC) [17] is used to perform the volumetric DoS attack against a secure version of OpenPLC running with the patched IPS. Various network parameters aree studied and the performance of the proposed embedded IPS system is analyzed in this section.

5.1 Deployment time

Three key steps determine the deployment time of proposed IPS. First, commissioning when the PLC is first connected to the operational network, the IPS collects the necessary data to train its initial model. Second, as more data is collected, the cluster centers of IPS are recalibrated or updated on the second iteration. Lastly, the model is deployed and the IPS starts monitoring the incoming network traffic. During this period of initial training and calibration, the initial data is assumed to be free of cyber-attacks and hence it is imperative to study the amount of time taken by the IPS system to get functional. In this analysis, the time taken to train the initial version of the IPS and the final deployment time of the IPS is plotted against the different scan times of the HMI. Figure 6 shows the deployment time of IPS for various scan times of the HMI. A typical industrial HMI queries the PLC for the status of the SCADA system every 50 - 500 milli seconds. These query times were considered while analyzing the deployment time of the IPS.

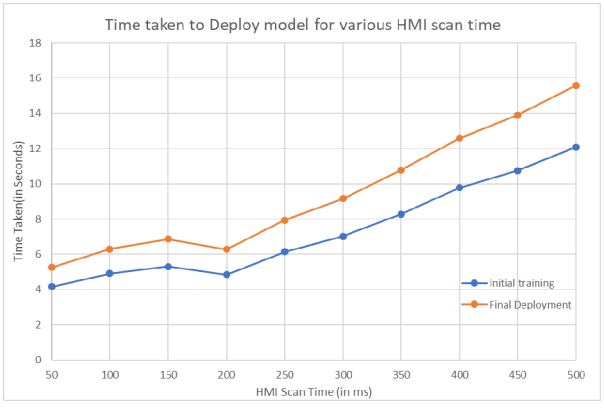

Figure 6: *Deployment time of the IPS*

The query rate or the scan time of the HMIs connected to the IPS determines how quickly the IPS can collect data from the live network. Figure 6 demonstrates the initial training time (time taken to deploy the model for the first time) and the final deployment time (Time taken by the IPS to start monitoring for cyber threats). The time taken between the initial training and the final deployment is the callibration time which is used by the IPS to change the cluster center to better match the conditions of the network.

5.2 IPS response Time

In this analysis, different parameters of the DoS attack were altered to study how quickly the IPS responds to a DoS attack.

5.2.1 Varying message size

This analysis reveals how much time the IPS takes to block the attacker once the DoS attack commences as demonstrated in figure 8. Different sizes of network packet payload are used to test the responsiveness of the IPS for various DoS attack scenarios. The number of attacker thread is fixed at 500 and the network packet payload size is varied from 200 to 2000 bytes. Figure 7 demonstrates the number of allowed requests to the PLC while Figure 8 shows the time taken by the IPS to block the attacker once the attack is started.

With increasing message length, the processing time taken by the PLC increases and since the network processing time is considered as an attribute in the machine learning module the IPS becomes more sensitive towards the cyber-attack. With the increase in network packet length, less number of malicious requests are allowed through the IPS to the PLC. This behavior of the IPS is demonstrated the Figure 8.

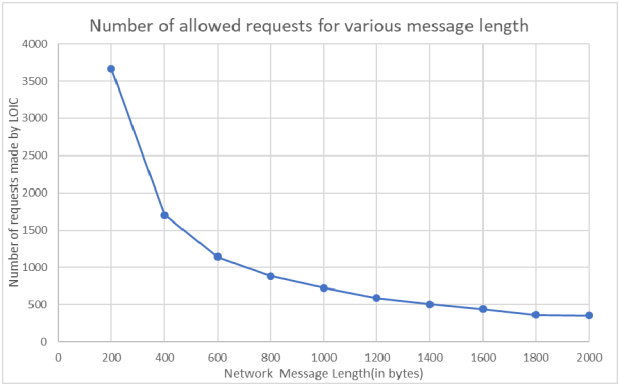

Figure 7: *Number of allowed requests for various message lengths during DoS attack using LOIC*

The time taken by the IPS to block the attacker after the DoS attack commences is found to be independent of the message length of the network packets. Hence the response of the IPS is not affected by changing the length of the packets being used by LOIC to flood the PLC. This observation is demonstrated in Figure 8.

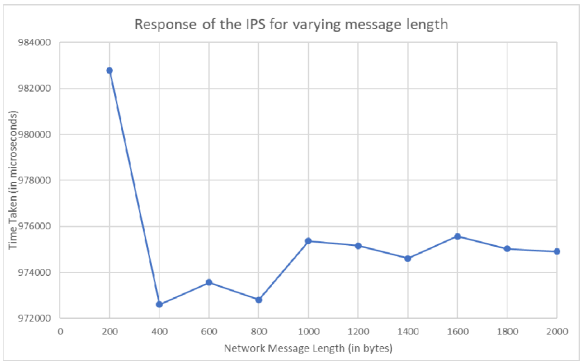

Figure 8: *Time taken by the IPS to block the attacker for various message lengths during DoS attack using LOIC*

5.2.2 Varying number of attacker threads

In this analysis, the size of the network packet payload is kept constant at 1200 bytes and the number of attacker thread is varied between 100 to 10000 to study the change in the response of the IPS during the DoS attack.

The number of DoS requests sent to the PLC before the attacker is blocked is demonstrated in figure 9 and is found to independent of the of the number of threads being used by the attacker to send the DoS requests.

Similarly, the response of the IPS is not affected by the number of thread the attacker uses for executing the DoS attack. This observation is demonstrated in Figure 10

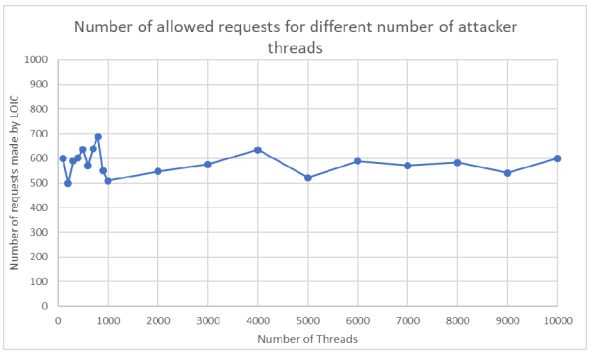

Figure 9: *Number of allowed requests for different number of threads during DoS attack using LOIC*

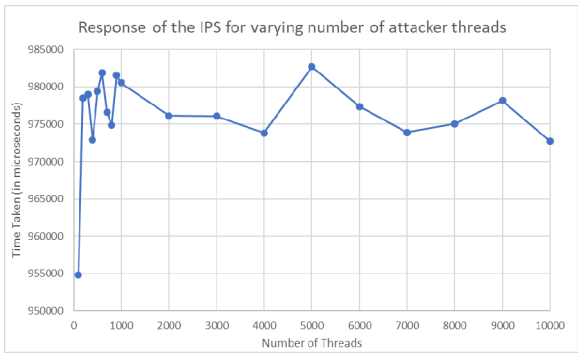

Figure 10: *Time taken by the IPS to block the attacker for various number of threads during DoS attack using LOIC*

5.3 IPS Recovery Time

After an attack has been detected, the incident response module blocks the attacker and restarts the network sockets. This causes a brief interruption in communication between the HMI and the PLC, but the huge number of network packets sent by the *Low Orbit Ion Cannon* is flushed out by the Operating System (OS). This enables the PLC to receive fresh queries from the HMI without going through every network packet sent by the LOIC and speeds up the recovery after the attack. Time taken by the IPS to refresh the network socket (IPS recovery time) is a very important parameter that needs to be analyzed to ascertain the performance of the IPS. Hence in this analysis the IPS recovery time is analyzed while the volumetric DOS attack is performed with varying message size and varying attacker threads. 100 runs with randomized message length and attacker threads yields an average recovery time of 6.3095 seconds. Hence the communication between the HMI and the PLC will be disrupted for 6.3095 sec while the IPS restarts the network socket.

6. CONCLUSION

The intrusion prevention system interfaces all packets received from the external network. It boasts off an implementation of an embedded unsupervised clustering algorithm to classify the incoming streaming data real-time to detect network anomaly and DoS attacks. If an attack is detected the IPS creates its own custom rules to block the attacker's IP from the network. The clustering algorithm (Bisecting K-means) makes the IPS generic and adjustable to any changing network condition. The detection and rule execution by the IPS efficiently prevent DoS attack from the trusted nodes.

Since the IPS resides inside the PLC the possibility of attackers compromising the capability of the IPS reduces significantly. Secondly, the embedded IPS adds one more security layer to the SCADA architecture thereby consolidating the defense-in-depth security approach. As a future work more, attributes can be considered in the machine learning module to detect and prevent a broader range of network attacks or anomalies.

7. REFERENCES

[1] Paganini, Threat Landscape Stakeholder Group. "The Number Of ICS Attacks Continues To Increase Worldwide". Security Affairs. N.p., 2017.

[2] C. Alcaraz, G. Fernandez and F. Carvajal, Security aspects of SCADA and DCS

environments, in Critical Infrastructure Protection, J. Lopez, R. Setola and S.

[3] Solum, Martin, “Quickdraw Retrospective, Part #1,” Digital Bond, November 17, 2009, http://www.digitalbond.com/2009/11/17/quickdraw-retrospective-part-1/; “Quickdraw Retrospective, Part #2,” Digital Bond, November 19, 2009, http://www.digitalbond.com/2009/11/19/quickdraw-retrospective-part-2/;

[4] Dulaunoy, A., Wagener, G., & Wagner, C. (2017). An extended analysis of an IoT malware from a blackhole network, (50), 13–15.

[5] Antonakakis, M., April, T., Bailey, M., Bursztein, E., Cochran, J., Durumeric, Z., … Yi Zhou, B. (2017). Understanding the Mirai Botnet. Proceedings of the 26th USENIX Security Symposium, 1093–1110. Retrieved from https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/antonakakis

[6] T. Alves , R. Das , & T. Morris, (2016). Virtualization of Industrial Control System Testbeds for Cybersecurity. Proceedings of the 2nd Annual Industrial Control System Security Workshop on - ICSS ’16, 10–14. http://doi.org/10.1145/3018981.3018988

[7] Alves T., Buratto, M., de Souza, F., Rodrigues, T., “OpenPLC: An open source alternative to automation,” in 2014 IEEE Global Humanitarian Technology Conference (GHTC), pp.585-589, Oct. 2014, doi: 10.1109/GHTC.2014.6970342

[8] Zhu, Bonnie, Anthony Joseph, and Shankar Sastry. “A taxonomy of cyber attacks on SCADA systems.” In Internet of things (iThings/CPSCom), 2011 international conference on and 4th international conference on cyber, physical and social computing, pp. 380-388. IEEE, 2011.

[9] Bronk Christopher, and Tikk-Ringas Eneken. “The Cyber Attack On Saudi Aramco”. http://dx.doi.org/10.1080/00396338.2013.784468

[10] Falliere, Nicolas, Liam O. Murchu, and Eric Chien. “W32. stuxnet dossier.” White paper, Symantec Corp., Security Response 5 (2011).

[11] Adhikari, U., Morris, T., Pan, S., Applying Hoeffding Adaptive Trees for Real-Time Cyber-Power Event and Intrusion Classification, IEEE Transactions on Smart Grid, doi: 10.1109/TSG.2017.2647778

[12] Adhikari, U., Morris, T. and Pan, S. (2017). Applying Hoeffding Adaptive Trees for Real-Time Cyber-Power Event and Intrusion Classification. IEEE Transactions on Smart Grid.

[13] Chinthaka, P., Premachandra, C. and Amarakeerthi, S. (2018). Effective natural communication between human hand and mobile robot using Raspberry-pi. 2018 IEEE International Conference on Consumer Electronics (ICCE).

[14] Mihal'ov, J. and Hulic, M. (2017). NFC/RFID technology using Raspberry Pi as platform used in smart home project. 2017 IEEE 14th International Scientific Conference on Informatics.

[15] Arthur, David, and Sergei Vassilvitskii, Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms, Society for Industrial and Applied Mathematics (2007)

[16] Zhang, D., Liu, L. and Feng, G. (2018). Consensus of Heterogeneous Linear Multiagent Systems Subject to Aperiodic Sampled-Data and DoS Attack. IEEE Transactions on Cybernetics, pp.1-11.

[17] Farina, P., Cambiaso, E., Papaleo, G. and Aiello, M. (2015). Understanding DDoS Attacks from Mobile Devices. 2015 3rd International Conference on Future Internet of Things and Cloud.

[18] Zhu, J., Wang, Y., Zhou, D. and Gao, F. (2018). Batch Process Modeling and Monitoring With Local Outlier Factor. IEEE Transactions on Control Systems Technology, pp.1-14.

[19] Nishat M., Inshil D., Zhou, D. and KiJoon C. (2017). Evolving Neural Network Intrusion Detection System for MCPS. 2018 20th International Conference on Advanced Communication Technology (ICACT)