Unsupervised Time-Series based Anomaly Detection in ICS/SCADA Networks

Ali Tekeoglu
Johns Hopkins University
Applied Physics Laboratory
ali.tekeoglu@jhuapl.edu

Korkut Bekiroglu
SUNY Polytechnic Institute
Electrical Engineering Technology
korkut.bekiroglu@sunypoly.edu

Chen-Fu Chiang
SUNY Polytechnic Institute
Computer Science
chiangc@sunypoly.edu

Sam Sengupta
SUNY Polytechnic Institute
Computer Science
sengupta@sunypoly.edu

Abstract—**Traditionally, Industrial Control Systems (ICS) have been operated as air-gapped networks, without a necessity to connect directly to the Internet. With the introduction of the Internet of Things (IoT) paradigm, along with the cloud computing shift in traditional IT environments, ICS systems went through an adaptation period in the recent years, as the Industrial Internet of Things (IIoT) became popular. ICS systems, also called Cyber-Physical-Systems (CPS), operate on physical devices (i.e., actuators, sensors) at the lowest layer. An anomaly that effect this layer, could potentially result in physical damage. Due to the new attack surfaces that came about with IIoT movement, precise, accurate, and prompt intrusion/anomaly detection is becoming even more crucial in ICS. This paper proposes a novel method for real-time intrusion/anomaly detection based on a cyber-physical system network traffic. To evaluate the proposed anomaly detection method's efficiency, we run our implementation against a network trace taken from a Secure Water Treatment Testbed (SWAT) of iTrust Laboratory at Singapore.**

Index Terms—**Anomaly Detection, Secure Water Treatment Testbed Dataset, Time-series, ICS SCADA Network Security.**

I. Introduction

The issue of Anomaly/outlier Detection (AD) in networking and various cyber-physical systems continues to be one major research question today. Various data types such as time-series, network packet-captures, and event-based data streams have been used in functionally diverse systems to detect anomalous event in real-time [1], [2]. Most results report anomaly detection from additional meta-data as conferred by machine learning-based methods.

Classical intrusion detection systems operating from a pre-recorded signature database, matching live traffic against known signatures to discover and alert on intrusions cannot work on new type of attacks. A failure to detect network intrusion is specifically unacceptable in industrial and supervisory control and data acquisition (SCADA) systems due to the potential catastrophic effects in physical world, or even human lives. Timely and accurate determination of both the known and unknown anomalies in such systems become vitally important for companies and countries. Each networking layer of an ICS system has to be monitored for anomalies; however, to prevent physical damage to the surroundings, the lowest layer where IIoT devices operate, needs extra attention, where lightweight cost-effective AD algorithms would be invaluable.

In this study, we propose an extension on the recently developed and reported unsupervised method [3] adapted towards ICS/SCADA network traffic AD. The proposed method is investigated for real-time intrusion/anomaly detection based on a cyber-physical system network packet trace captured from a realistic miniature water treatment testbed, under normal operation as well as from potential attacks. The network traffic features are used to capture the system-specific anomalies. First, these features are investigated in network packet traces, and they are converted to time-series data by a method similar to the time-windowing approach described in [4]. The generated time-series data set is used as an input to our AD method. We used Secure Water Treatment (SWaT) dataset [5], [6] to test the proposed algorithm. SwaT consists of a six-stage water treatment process and contains many sensors, actuators, and chemical processes. A network packet capture from all the computers, devices, motors, and sensors has been collected in this data set, during normal operation and attack conditions. The details about the data-set are given in section III.

The rest of the paper is organized as follows. In section II, the recent literature on anomaly detection in SCADA networks is summarized. Section III provides the details about the SWaT dataset and its infrastructure. Section IV gives the details about the methods used in this paper. In section V, the performance of the proposed work is presented. Finally, the future direction and conclusion are given in the section VI.

II. Related Work

An integrated Internet of Things (IoT) platform enables a promising way to construct robust and cyber-attack proof industrial systems and applications [7]. Cyber-attacks and intruders often aim at obtaining sensitive data that sensors and actuators acquire, analyze, and manage. Once the data collected are sent to the Cloud, a Cloud-based architecture for the Internet of Things (IoT) applications can be adapted to improve smart industrial systems' deployment and provide remote monitoring and control [8]. As a result, ICS and its components such as Supervisory Control and Data Acquisition

The Python implementation for methods described in this paper are shared on GitHub and can be reached at https://git.io/JqFIo. Please feel free to reach authors for any questions comments.

(SCADA) systems become vitals targets of various cyber-attacks [9]. Several successful attacks have been carried out in the last decade, and the frequency tends to increase. The security of ICS and SCADA systems is a vital research area for researchers [10], [11]. Even though various AD methods have been developed and investigated, malicious attacks continue. Researchers began investigating semi-supervised approaches to automatically integrate a new attack to the database [12]. For instance, the study in [13] combined the feature reduction and detection methods for the smart grid to improve the anomaly detection accuracy while diminishing the computational complexity for a supervised method. The proposed method does not require an attack database. Instead, the model dynamics are updated continuously to detect any system behavior changes. However, real-time detection with high accuracy remains challenging without an attack database.

Moreover, various data types and structures have been tested for a different type of AD algorithms. For instance, temporal data provide the sequential behavior of network traffic. It has been useful to use temporal data to detect crucial anomalies in many fields, such as IT, finance, and medical domains. For online data where the network traffic is in real-time, it is hard to have the correct label at the instance of the occurrence, and there might exist some temporal correlations among features that introduce the anomaly. The scalability and portability issues are other hurdles for many state-of-the-art unsupervised machine-learning techniques for anomaly detection. Recently the work on TadGAN, an unsupervised anomaly detection approach built on Generative Adversarial Networks (GANs), captures the temporal correlations of time series distributions using the Long Short-Term Memory (LSTM) approach for Generators and Critics [14].

In general, SCADA systems with embedded IoT technology, complex and controlled centrally, are susceptible to external attacks leading to anomalies. A decentralized system architecture to improve a SCADA system's security is proposed in [15]. The study in [15] investigated the challenges coming from their centralized structure and then proposed a method by using the tamper-resistant ledger, still requiring a training dataset and an attack database. *Our proposed method also can be decentralized and implemented in a small device because of its computational efficiency.*

Detecting and preventing such malicious activities requires continuous monitoring and recording of new attacks in a database. Updating the database prevents real-time anomaly detection for ICS for a new attack. Also, scientists have been investigating computationally efficient semi-supervised methods to improve the security [12]. However, these methods still require various steps, such as the training process. Some deep-learning-based anomaly detection approach was proposed in [10]. Even though the proposed methods in [10] is an adaptive detection approach, this method still has the same limitation, requiring a big-data to train the proposed model. *We propose a computationally efficient unsupervised anomaly detection scheme in this paper to address the limitations of historical data requirements. Since the proposed method is computationally tractable, it can be applied to the IoT device or sensor level.*

III. Background on SWaT Test-bed and Dataset

Secure Water Treatment (SWaT) test-bed is an industrial control system developed and operationalized by Singapore University of Technology and Design [16]. The SWaT test-bed is utilized to test various cyber-physical attacks against the components of the system. The test-bed can also be used to assess the effectiveness of anomaly detection methods against these attacks. There are six stages (P1 - P6) in the test-bed in water treatment, and a programmable logic controller (PLC) unit controls each of them. The details of these stages can be foound in [16].The wired and/or wireless communication between components such as sensors, actuators, PLC, etc., is performed through a local Fieldbus. The Historian stores the real-time data, and in this study, we will use this network traffic data generated under various cyber-physical attacks. The malware was a custom malware developed by the researchers in this test-bed [5].

SWaT test-bed was run 11 days of continuous operation, and with various time intervals, a total of 7 days' data was collected under normal operation, and a total of 4 days' data was collected under various cyber-physical attacks. The sensors' measurements, control signals, actuators, network traffic, and other components data were recorded. The data that were tested in this paper by the proposed method were collected in December 6th, 2019. Each data is labeled as –a normal & attack. The duration of malware to the SCADA was a 5 min attack and a 10 min sleep. Also, the disturbance to the sensor readings was a 3 min attacks and a 10 min sleep (see Table I). This pattern was cycled during the data collection.

The anomaly detection method described in this paper is based on network traffic captures; thus, we utilized only the PCAP files of the SWAT Dataset. We wanted to focus on one day of the SWAT dataset, which recorded network traffic from 10:05 am to 13:45 pm (about 3.5 hours) on December 6th. SWAT Dataset (PCAP files) were split into individual PCAP files that contain 15 minutes intervals each. The time intervals and events during the intervals are listed in Table I.

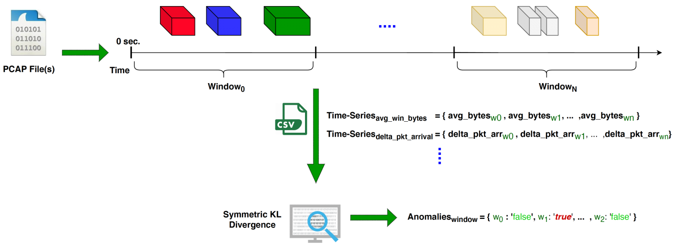
PCAP File(s)
0 sec.
Time
Window$_0$
Window$_N$
Time-Series$_{avg\_win\_bytes}$ = { avg_bytes$_{w0}$, avg_bytes$_{w1}$, ... ,avg_bytes$_{wn}$ }
Time-Series$_{delta\_pkt\_arrival}$ = { delta_pkt_arr$_{w0}$, delta_pkt_arr$_{w1}$, ... ,delta_pkt_arr$_{wn}$}
Symmetric KL Divergence
Anomalies$_{window}$ = { w$_0$: 'false', w$_1$: 'true', ... , w$_2$: 'false' }

Fig. 1: Anomaly Detection Process Flow Diagram

IV. Tools and Methods

Network packet captures provide an immense amount of information. Communication between each pair of the device is divided into network packets, where each packet contains

Time Interval	Event	Label Assigned	Time Interval	Event	Label Assigned
`10:05-10:20`	Pre-attack, no anomalous events, idle network traffic.	*No-Attack*	`10:20-10:30`	Infiltrate SCADA Workstation with USB Malware injection, no network effect yet.	*No-Attack*
`10:30-11:30`	Ex-filtrate Historian Data (5 mins attack + 10 mins sleep) x 4-cycles	*Attack (5min) & No-Attack (10min)*	`11:30-12:30`	No anomalous events, idle network traffic.	*No-Attack*
`12:30-12:30`	Infiltrate SCADA WS with second malware, via downloading from C2 server.	*Attack*	`12:30-13:30`	Disrupt sensor readings and process (3 mins attack + 10 mins sleep) x 5-cycles	*Attack (3min) & No-Attack (10min)*
`13:30-13:45`	Capture post-attack pcap for 15 minutes.	*No-Attack*			

TABLE I: Events, time intervals and our labels in SWAT dataset.

meta-data (i.e., packet headers) and a payload. Insecure networks, the payload is encrypted; however, one can still infer insights from the meta-data contained in each packet. In its raw form, network captures do not contain information in time-series form. In a given network, sometimes there might be a huge amount of traffic, while other times, there might be no traffic at all. The frequency of the packets traveling in the network is almost never uniform, whereas time-series data would provide samples within a repeating interval.

Thus, we designed an approach to convert network traffic capture into time-series data. Our method was inspired by a paper [4] in which researchers were developing techniques for the Internet of Things botnet detection. The proposed approach, depicted in Figure 1, starts with converting network packet captures into time-series data. Generated time-series data in CSV format are then feed into our main algorithm 1. An anomaly detection algorithm operates on overlapping, sequential windows, which contain multiple data-points. Running averages and probabilistic distances between sequential windows give us anomaly scores that is calculated by the algorithm given in algorithm 2. Our algorithm would report anomalous windows in the time-series data-set at the end.

A. Method to Convert PCAPs to Time-Series

PCAP files are network traffic recordings, consisting of each and every packet traversing the network over the wire (or wirelessly). We wanted to extract meaningful packet based features from PCAP files in time-series format, that would signal the anomalies in the network. For this purpose, we start with a time-window (i.e. 3 seconds), divide the network packets falling into each time-window, calculate specific features from the packets in a window, and output the calculated features in a comma-separated-list. This process is explained in Figure 1.

1) Feature Development: In a PCAP file for each window, there are potentially multitudes of statistical features that could be calculated which would in turn be used as time-series data for anomaly detection. However, choosing the most valuable statistical feature is critical for the precision of the detection algorithm. We have extracted the parameters from each time window, that are listed in Table II, as potential futures from the network traffic captures. Feature engineering for the most effective parameters are based on [17].

Variable	Definition
`AvgBytes`	Average bytes in the window
`SumETHBytes`	Total number of bytes in Ethernet frames in the window
`WinPktCnt`	Total number of packets in the window
`NumSrcIPs`	Total number of unique source IP addresses in the window
`NumDstIPs`	Total number of unique destination IP addresses in the window
`NumNonIPv4Pckts`	Number of Non IPv4 Packets in the window (i.e. IPv6, ARP, LLC)
`AvgInterPktTime`	Average seconds between each consecutive packets in a window

TABLE II: Potential futures/parameters for Time-Series generation from PCAP files

B. Symmetric Kullback-Leibler (KL) Divergence Based Anomaly Scoring

An unsupervised anomaly detection method [3] is proposed and tested with a cloud intrusion detection dataset from the University of Victoria, called "ISOT-Dataset". The data-set included cloud hypervisor heart-beat information (as a time-series) in the vmstat command output with 11-second intervals. We focused on detecting anomalies in CPU utilization information. In follow-up work, we employed a special matrix called *Hankel Matrices* [18] to exploit its properties, which in turn enabled us to detect anomalies with a significant accuracy efficiently. In work, [3], the distribution of time series data is calculated for given time windows. Then the distribution computed for a given window is then compared with the Kullback-Leibler divergence measure. Since we improved the same method for SCADA systems in this work, it is summarized in this section.

Initially, the probability distribution of time series data κ for a given window is calculated with *Softmax function*[1]. The SoftMax function is defined as

$$\mathcal{S}(\kappa) = \frac{e^{-\kappa_i}}{\sum_{j=1}^{N} e^{-\kappa_j}} \tag{1}$$

where vector $\kappa \in \mathbb{R}^N$ and $\mathcal{S}(\kappa) \in \mathbb{R}^N : \mathcal{S}(\kappa_i) \in [0,1]$. Furthermore, to measure the differences between the two distributions, we used the Kullback-Leibler (KL) divergence[2]. The KL divergence of a distribution P with respect to a distribution Q is represented as;

$$D_{KL}(P||Q) = \sum_{x \in X} P(x) log \frac{P(x)}{Q(x)} \tag{2}$$

[1]The softmax function converts the real values in a vector K to a K_{new} real numbers such that $\sum K_{new} = 1$ and $K_{new} \geq 0$.

[2]KL divergence [19] is a measure that represents how two distributions, discrete or continuous, differ from each other.

where $P(x)$ is the x^{th} element of vector P. Note that, $D_{KL}(P||Q) \geq 0$, with equality only if $P = Q$. In general, relative entropy is not symmetric, meaning interchange of the distributions P and Q would result in a different distance, i.e.

$$D_{KL}(P||Q) \neq D_{KL}(Q||P) \tag{3}$$

Method in [3], an active window w and its corresponding elements from time series data is recorded. Also three consecutive window before $w \in \mathbb{R}^n$, $w_i \in \mathbb{R}^n,\ i = 1, 2, 3$ is recorded from the same time series data. Then using the Softmax function, the probability distribution of each corresponding vector is calculated such that $\mathcal{S}(w_i),\ i = 1, 2, 3$. Finally the distribution of active window w is calculated such that $\mathcal{S}(w)$.

After calculating the probability distribution of the past three consecutive window and active window, the linear combination of three preceding historical windows is combined such that the effect of the closest window to the active window is the highest. For this purpose, three new coefficient parameters are defined, $m_3 < m_2 < m_1$, s.t. $m_1 + m_2 + m_3 = 1$. Based on these distributions and given parameters, the anomaly score, $f(w_i)$, of the current window, w_i, is calculated as follows for window indices $i \geq 3$,

$$\begin{aligned} f(w_i) = D_{KL}\Big(&\mathcal{S}(w_i) \mid\mid m_1\mathcal{S}(w_{i-1}) + m_2\mathcal{S}(w_{i-2}) + m_3 \\ &\mathcal{S}(w_{i-3})\Big) + D_{KL}\Big(m_1\mathcal{S}(w_{i-1}) + m_2\mathcal{S}(w_{i-2}) \\ &+ m_3\mathcal{S}(w_{i-3}) \mid\mid \mathcal{S}(w_i)\Big) \end{aligned} \tag{4}$$

Via interchanging the position of two input distributions in (4), we eliminated the asymmetrical behaviour of KL-Divergence as shown in (3), and achieved a true distance metric between two probability distributions.

Anomaly score calculations for the first 3 windows ($w_i \mid 0 \leq i < 3$) requires a somewhat special treatment, since they don't have 3 preceding windows. Anomaly score for first window (w_0) is defined as

$$f(w_0) = D_{KL}(\mathcal{S}(w_0) \mid\mid \mathcal{S}(w_0)) = 0 \tag{5}$$

While, the second window's (w_1) anomaly score is calculated as follows

$$f(w_1) = D_{KL}\Bigg(\mathcal{S}(w_1) \mid\mid m_1\mathcal{S}(w_0)) \ + \ D_{KL}(m_1\mathcal{S}(w_0) \mid\mid \mathcal{S}(w_1)\Bigg) \tag{6}$$

Similarly, the third window's anomaly score, $f(w_2)$, is calculated from the weighted symmetric KL distance of w_2 with respect to preceding windows, w_1 and w_0.

Furthermore, to minimize the effect of noises on the anomaly score, the classical moving average process is employed. For each sliding time window w_i, with a set parameter γ, the mean μ_{w_i} and standard deviation σ_{w_i} are repeatedly updated by the following equations;

$$\begin{aligned} \mu_{w_i} &= \gamma\mu_{w_{i-1}} + (1-\gamma)f(w_i) \\ &\quad \text{for } i = 0,\ \mu_{w_i} = 0 \\ \sigma_{w_i} &= [\gamma\sigma^2_{w_{i-1}} + (1-\gamma)(f(w_i) - \mu_{w_i})^2]^{\frac{1}{2}} \\ &\quad \text{for } i = 0,\ \sigma_{w_i} = 0 \end{aligned} \tag{7}$$

Finally this moving average and anomaly score $f(w_i)$ in (4) are compared to detect the anomalies in real-time. Given a fix parameter α, we used

$$\underbrace{f(w_i) - \overbrace{(\mu_{w_{i-1}} + \alpha\sigma_{w_{i-1}})}^{\psi}}_{\Delta} > \varepsilon \tag{8}$$

The overall algorithm that we have designed for anomaly detection on time-series data points, as detailed in work [3] is summarized in listing Algorithms 1 and 2.

Algorithm 1 Symmetric KL Divergence Anomaly Detection

```
procedure SYMKL(in: $csv$, out: $anomalies$)
    $windowSize \leftarrow$ grid search (see Table III)
    $windowShift \leftarrow$ grid search (see Table III)
    $nBinsPerWndw \leftarrow$ grid search (see Table III)
    $r \leftarrow csv.nextRow()$
    while $r \neq Null$ do        ▷ till all CSV lines are read
        $tmstmp \leftarrow r.time\_stamp\_column\_index()$
        $data \leftarrow r.time\_series\_column\_index()$
        $lst\_cur\_window \leftarrow \{tmstmp, data\}$    ▷ append
        if $length(lst\_cur\_window) == 1$ then
            $win\_start \leftarrow tmstmp$
            $win\_end \leftarrow tmstmp + windowSize$
        end if
        while $time\_stamp < win\_end$ do
            $r \leftarrow csv.nextRow()$
            $tmstmp \leftarrow r.time\_stamp\_column\_index()$
            $data \leftarrow r.time\_series\_column\_index()$
            $lst\_cur\_window \leftarrow \{tmstmp, data\}$  ▷ apnd
            calc. bin distribution for window.
            calc. Anomaly Score with Algo 2 for window.
        end while
        if $time\_stamp \geq win\_end$ then
            $win\_start \leftarrow win\_start + windowSlide$
            $win\_end \leftarrow win\_start + windowSize$
            $lst\_cur\_window \leftarrow \{\}$ ▷ clear current windw
            $r \leftarrow csv.nextRow()$
        end if
    end while
    return $lst\_anomalies$ ▷ Time-windows w/Anomalies
end procedure
```

V. Experiments and Results

The proposed method of network based anomaly detection requires several parameters, as explained in previous section IV. Most of the parameters have to be fine-tuned to get the best precision. In order to find the optimum parameter values, we implemented a hyper-parameter search function, and experimented with range of potential values for variety parameter combinations. List of parameters along with the range of values we have tested to discover optimal anomaly detection performance of our method are listed in Table III.

After searching for each combination of the parameter's range of values (i.e. 15120 combinations), we calculated True

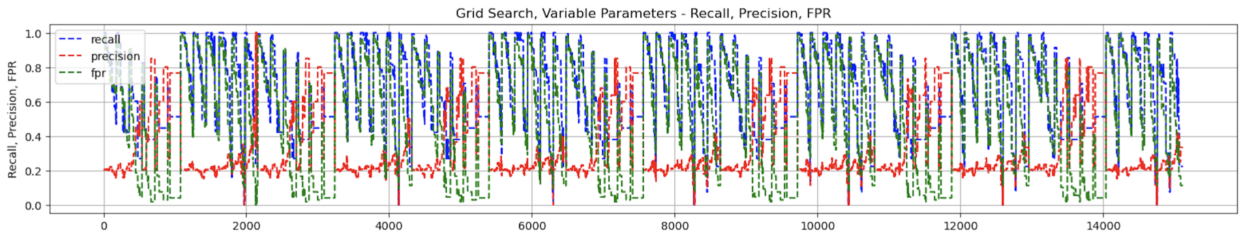

Fig. 2: Grid search for parameter tuning, on time-series generated with **0.5** second time interval on PCAP sliding-windows. The range of parameters are defined in Table III.

Algorithm 2 Anomaly Score Calculation Algorithm

Inputs: $\{m_3 < m_2 < m_1 \text{ s.t } m_1 + m_2 + m_3 = 1\}$, γ, α, ε, size of each window n, sliding data size l, anomaly score threshold ε, csv (time series data).

```
    Initiate: $\sigma_{w_{i-1}} = 0$, $\mu_{w_{i-1}} = 0$, $f(w_{i-1}) = 0$, $\theta = 1$, $AFlag$ =False
    repeat
 1:  Update vectors $w_{i-3}, w_{i-2}, w_{i-1}$ and $w$ with the data in $csv$.
 2:  Calculate $\mathcal{S}(w_{i-1})$, $\mathcal{S}(w_{i-2})$, $\mathcal{S}(w_{i-3})$, $\mathcal{S}(w)$ (distributions using (1))
 3:  Find anomaly score $f(w_i)$ using (4)
 4:  Calculate $\mu_{w_i}$ and $\sigma_{w_i}$ using (7)
 5:  if $f(w_i) - (\mu_{w_{i-1}} + \alpha\ \sigma_{w_{i-1}}) \geq \varepsilon$ & $AFlag$ =False then
 6:      $AFlag$ =True and Anomaly Detected!      ▷ Entering the anomaly region
 7:  end if
 8:  if $f(w_i) - (\mu_{w_{i-1}} + \alpha\ \sigma_{w_{i-1}}) \geq \varepsilon$ & $AFlag$ =True then
 9:      Anomaly Detected!      ▷ In anomaly region
10:  end if
11:  if $f(w_i) - (\mu_{w_{i-1}} + \alpha\ \sigma_{w_{i-1}}) < \varepsilon$ & $AFlag$ is True then
12:      Go to Initiate
13:  end if
14:  if $f(w_i) - (\mu_{w_{i-1}} + \alpha\ \sigma_{w_{i-1}}) < \varepsilon$ then
15:      No Anomaly
16:      $\sigma_{w_{i-1}} = \sigma_{w_i}$, $\mu_{w_{i-1}} = \mu_{w_i}$, $f(w_{i-1}) = f(w_i)$
17:  end if
```

Parameter	Range of Values
`WindowSize`	[20, 40, 60, 80, 100, 120]
`WindowShift`	$WindowSize * 0.2$
`NumofBins`	10, 20
m_1	[0.5 , 0.55, 0.6 , 0.65, 0.7 , 0.75, 0.8]
m_2	$[(1 - m_1) * 0.6]$
m_3	$[(1 - m_1) * 0.46]$
α	[0.4 , 0.48, 0.56, 0.64, 0.72, 0.8]
γ	[0.1, 0.2, 0.3, 0.4, 0.5]
ϵ	[0.0025, 0.0175, 0.0325, 0.0475, 0.0625, 0.0775]

TABLE III: Set of values tested for parameters of the anomaly detection algorithm ($6*2*7*6*5*6 = 15120$ combinations). Out of 15k runs, optimum parameter values with respect to anomaly detection accuracy is given in Table IV.

Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN) values, according to the time-stamps and labels listed in I. In Machine Learning domain, there are several popular statistical methods for determining performance of a classification algorithm, based on TP/FP/TN/FN values. True Positive Rate (TPR, also called; Recall, Sensitivity, Hit Rate or Probability of Detection) and False Positive Rate (FPR, also called; fall-out, or false alarm ratio) are calculated as follows; $TPR = TP/(TP+FN)$, $FPR = FP/(FP+TN)$.

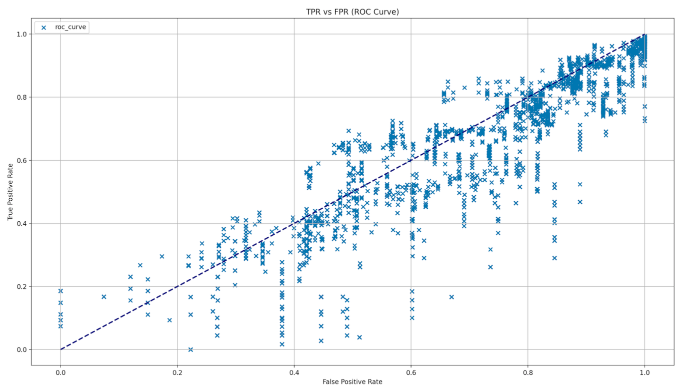

Fig. 3: TPR vs FPR (R.O.C. curve) for 15k runs with variable parameters. TPR-FPR values above the diagonal line represent better than random detection.

TPR and FPR are used to plot the R.O.C. curve depicted in 3. In addition to TPR and FPR, we also calculated Accuracy and F1-score for each run of the experiment, which are formulated as follows; $Accuracy = (TP+TN)/(TP+TN+FP+FN)$, $Precision = TP/(TP+FP$, $F_1\ score = 2*(Precision*Recall)/(Precision+Recall)$.

Grid search results with the parameter ranges are given in the Table III, and also plotted in Figure 2. The best accuracy and F1score values are given in Table IV.

A. *Discussion*

Traditional machine learning algorithms utilize a diverse set of futures representing several dimension (i.e. columns of the train/test data-set). Deep learning models built with neural networks learn many features from a given input data-set. With a single time-series parameter, it proved to be harder to model single, super-feature that would represent different types of network attacks. In the data-set we used, there are 1 non-network and 3 types of network based attacks; (i) Infection

TP	FP	TN	FN	TPR	FPR	Accuracy	F1score
552	169	4151	527	0.512	0.039	0.871	0.613
481	120	4200	598	0.446	0.0278	0.868	0.573
409	72	4248	670	0.380	0.017	0.863	0.524
409	73	4247	670	0.379	0.017	0.862	0.524
529	194	4126	550	0.490	0.045	0.862	0.587
529	314	4006	550	0.490	0.072	0.840	0.550

TABLE IV: Results are sorted by `Accuracy`, `F1-score`, `True Positive Rate (TPR)` respectively in decreasing order, to get best detection rates out of 15k runs with varying parameter combinations. Parameter values resulting in the best performance shown in the first row above, indicating best experimental results are as follows; m_1: 0.5, m_2: 0.3, m_3: 0.2, `NumofBins`: 10, `WindowSize`: 120, `WindowShift`: 24, ϵ: 0.0175, γ: 0.1, α: 0.4.

of a workstation with a USB drive (no immediate network anomaly during infection), (ii) database ex-filtration from infected local computer to a remote attacker, (iii) malware download from a remote C&C server, (iii) attacks from an infected internal machine to local sensors. All these attacks (except USB infection) have different characteristics in the network traffic, so time-series data generated from a single network traffic feature results in over-generalization. Even with these disadvantages, our approach performed 87.11% Accuracy, 61.33% F1-Score with the best parameter values. For further experimentation, we are going to work on designing better features, and use a weighted combination of improved features to generate time-series data for anomaly detection algorithm to operate on. On the positive side, since our method is computationally light-weight and efficient, running our algorithm on multiple processes, operating on time-series data generated from different features, each tuned for catching different types of network attacks would significantly improve overall anomaly detection rate of the system. Each system has a different tolerance for false-positives and false-negatives. We have tuned our parameters for highest Accuracy, F1-score and Precision while having lowest Fall-Out rate. We believe these combination of scores be suitable for a network anomaly detection system in ICS networks.

VI. Conclusion and Future Work

Lightweight and real-time anomaly detection in ICS and Critical Infrastructure domain is crucial due to the potential physical effects of undetected attacks on these networks. We have developed an unsupervised, efficient anomaly detection method that processes network traffic and operates on parsed time-series data. Even though our approach is effective, the crafting of the most appropriate metric is required. Thus, we investigated a dataset captured from a real water treatment testbed, which included two types of attacks on the Historian server as well as disruption of sensor readings. Our initial findings show that a good accuracy in terms of true anomaly detection. Future work will include improving the accuracy, computational optimizations, and investigation of better metrics to choose for time-series generation from network captures specific to attacks against ICS/SCADA networks [3].

[3]Authors would like to thank SUNY Polytechnic Institute and graduate student Harsh Thakkar for his invaluable feedback and support

References

[1] A. Blázquez-García, A. Conde, U. Mori, and J. A. Lozano, "A Review on Outlier/Anomaly Detection in Time Series Data," *arXiv preprint arXiv:2002.04236*, February 2020. [Online]. Available: https://arxiv.org/pdf/2002.04236.pdf

[2] M. Fahim and A. Sillitti, "Anomaly Detection, Analysis and Prediction techniques in IoT Environment: A Systematic Literature Review," *IEEE Access*, vol. 7, pp. 81 664–81 681, 2019.

[3] B. Andriamanalimanana, A. Tekeoglu, K. Bekiroglu, S. Sengupta, C. Chiang, M. Reale, and J. Novillo, "Symmetric Kullback-Leibler Divergence of Soft-Maxed Distributions for Anomaly Scores," in *2019 IEEE Conference on Communications and Network Security (CNS)*, June 2019, pp. 1–6.

[4] R. Doshi, N. Apthorpe, and N. Feamster, "Machine Learning DDoS Detection for Consumer Internet of Things Devices," in *2018 IEEE Security and Privacy Workshops (SPW)*, May 2018, pp. 29–35.

[5] Singapore University of Technology and Design, iTrust Lab, "Secure Water Treatment (SWaT) Test-Bed," https://itrust.sutd.edu.sg/itrust-labs-home/itrust-labs_swat/, accessed: Jan, 2021.

[6] S. Adepu and A. Mathur, "Distributed Attack Detection in a Water Treatment Plant: Method and Case Study," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 1, pp. 86–99, Jan 2021.

[7] L. D. Xu, W. He, and S. Li, "Internet of Things in Industries: A Survey," *IEEE Transactions on Industrial Informatics*, vol. 10, pp. 2233–2243, 11 2014.

[8] A. F. da Silva, R. L. Ohta, M. N. dos Santos, and A. P. Binotto, "A Cloud-based Architecture for the Internet of Things targeting Industrial Devices Remote Monitoring and Control," *IFAC-PapersOnLine*, vol. 49, pp. 108–113, 2016.

[9] H. A. Kholidy, A. Tekeoglu, S. Iannucci, S. Sengupta, Q. Chen, S. Abdelwahed, and J. Hamilton, "Attacks Detection in SCADA Systems using an Improved Non-Nested Generalized Exemplars Algorithm," in *2017 12th International Conference on Computer Engineering and Systems (ICCES)*, 2017, pp. 607–612.

[10] A. O. Gomez Rivera and D. K. Tosh, "Towards Security and Privacy of SCADA Systems Through Decentralized Architecture," in *2019 International Conference on Computational Science and Computational Intelligence (CSCI)*, Dec 2019, pp. 1224–1229.

[11] M. Cheminod, L. Durante, and A. Valenzano, "Review of Security Issues in Industrial Networks," *IEEE Transactions on Industrial Informatics*, vol. 9, pp. 277–293, 2013.

[12] G. Bernieri, M. Conti, and F. Turrin, "Evaluation of Machine Learning Algorithms for Anomaly Detection in Industrial Networks," in *2019 IEEE International Symposium on Measurements&Networking (M&N)*, July 2019, pp. 1–6.

[13] S. Huda, S. Miah, M. M. Hassan, R. Islam, J. Yearwood, M. Alrubaian, and A. Almogren, "Defending Unknown Attacks on Cyber-Physical Systems by Semi-Supervised Approach and Available Unlabeled Data," *Information Sciences*, vol. 379, pp. 211–228, 2 2017.

[14] A. Gumaei, M. M. Hassan, S. Huda, M. R. Hassan, D. Camacho, J. D. Ser, and G. Fortino, "A Robust Cyberattack Detection Approach using Optimal Features of SCADA Power Systems in Smart Grids," *Applied Soft Computing Journal*, vol. 96, 11 2020.

[15] S. Huda, S. Miah, J. Yearwood, S. Alyahya, H. Al-Dossari, and R. Doss, "A Malicious Threat Detection Model for Cloud Assisted Internet of Things (CoT) Based Industrial Control System (ICS) Networks Using Deep Belief Network," *Journal of Parallel and Distributed Computing*, vol. 120, pp. 23–31, 10 2018.

[16] A. P. Mathur and N. O. Tippenhauer, "SWaT: A Water Treatment Testbed for Research and Training on ICS Security," in *2016 International Workshop on Cyber-physical Systems for Smart Water Networks (CySWater)*, April 2016, pp. 31–36.

[17] I. Sharafaldin, A. H. Lashkari, and A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," in *4th International Conference on Information Systems Security and Privacy (ICISSP'18)*, 2018.

[18] K. Bekiroglu, A. Tekeoglu, B. Andriamanalimanana, S. Sengupta, C. Chiang, and J. Novillo, "Hankel-based Unsupervised Anomaly Detection," in *2020 American Control Conference (ACC)*, July 2020, pp. 5139–5144.

[19] S. Kullback and R. A. Leibler, "On Information and Sufficiency," *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79–86, 1951. [Online]. Available: http://www.jstor.org/stable/2236703