Received November 15, 2019, accepted December 7, 2019, date of publication December 18, 2019,
date of current version December 30, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2960497

Analysis of Machine Learning Methods in EtherCAT-Based Anomaly Detection

KEVSER OVAZ AKPINAR, (Member, IEEE), AND IBRAHIM OZCELIK
Department of Computer Engineering, Sakarya University, 54050 Sakarya, Turkey

Corresponding author: Kevser Ovaz Akpinar (kovaz@sakarya.edu.tr)

This work was supported in part by the BAPK, Scientific Research Projects Unit of the Sakarya University under Grant 2015-50-02-025, and in part by the TUBITAK, Scientific and Technological Research Council of Turkey under Grant 118E263.

ABSTRACT Today, the use of Ethernet-based protocols in industrial control systems (ICS) communications has led to the emergence of attacks based on information technology (IT) on supervisory control and data acquisition systems. In addition, the familiarity of Ethernet and TCP/IP protocols and the diversity and success of attacks on them raises security risks and cyber threats for ICS. This issue is compounded by the absence of encryption, authorization, and authentication mechanisms due to the development of industrial communications protocols only for performance purposes. Recent zero-day attacks, such as Triton, Stuxnet, Havex, Dragonfly, and Blackenergy, as well as the Ukraine cyber-attack, are possible because of the vulnerabilities of the systems; these attacksare carried by the protocols used in communication between PLC and I/O units or HMI and engineering stations. It is evident that there is a need for robust solutions that detect and prevent protocol-based cyber threats. In this paper, machine learning methods are evaluated for anomaly detection, particularly for EtherCAT-based ICS. To the best of the author's knowledge, there has been no research focusing on machine learning algorithms for anomaly detection of EtherCAT. Before testing anomaly detection, an EtherCAT-based water level control system testbed was developed. Then, a total of 16 events were generated in four categories and applied on the testbed. The dataset created was used for anomaly detection. The results showed that the k-nearest neighbors (k-NN) and support vector machine with genetic algorithm (SVM GA) models perform best among the 18 techniques applied. In addition to detecting anomalies, the methods are able to flag the attack types better than other techniques and are applicable in EtherCAT networks. Also, the dataset and events can be used for further studies since it is difficult to obtain data for ICS due to its critical infrastructure and continuous real-time operation.

INDEX TERMS Anomaly detection, EtherCAT security, ICS security, machine learning for EtherCAT.

I. INTRODUCTION

To ensure sustainability and maintain security, critical infrastructure networks need to be operated and monitored continuously. The critical infrastructure assets that provide this structure are called industrial control systems (ICS), and control of ICS is provided by supervisory control and data acquisition (SCADA) systems. Since the components of ICS applications have different requirements according to the location in which they are used in automation, ICS need to follow a defined hierarchical model. Previously, an automation hierarchy or computer-integrated manufacturing reference model was used.

Later, this became a modern structure known as the Purdue model, where the computer-integrated manufacturing reference model levels were divided into zones and security parameters were added [1], [2]. The Purdue model set out to transform ICS, which were previously isolated networks, into more robust structures, with protection against the potential cyber threats that ICS face due to the recent integration of operational technologies (OT) and information technologies (IT). NIST recommends that security in the Purdue model can be enhanced by adding monitoring systems such as intrusion detection/prevention systems (IDS/IPS), security information and event management (SIEM) software, and log aggregators, and by placing firewalls at all levels as a zone access point for inter-regional communication and external access to the zone [3].

The associate editor coordinating the review of this manuscript and approving it for publication was Ana Lucila Sandoval Orozco.

TABLE 1. Important cyber-events from past to present [4]–[9].

Attack	Year	Location	Result
Trojan attack of the gas pipeline (first known cyber-attack)	1982	Siberia	TNT explosion
Leak to company computers	1992	Chevron Emergency Alarm System - California, USA	Turning off the alarm system
Oil pipeline controller failure	1992	Natural Gas Company - Texas, USA	Material losses and death
Trojan via dialup modem to the billing system of the Salt River Project	1994	Phoenix, USA	Admin privileges, logs, passwords and data manipulation
Leak to the phone system	1997	MA, USA	Exclusion of important lines for 6 hours
Trojan attack of the natural gas pipeline	1999	Russia	Changing the gas control
SCADA system failure of oil storage unit	1999	England	Pollution as a result of explosion and deaths
Configuration fault in oil pipeline infrastructure	1999	Washington, USA	Pipeline explosion and deaths
Access to the waste management system via wireless radio	2000	Australia	Spread of raw waste to the environment
Leak to the SQL system	2003	CSX Company - Florida, USA	Switching off signals and alarms of the transportation system
Worm attack of the nuclear plant	2003	Ohio, USA	Disabling two monitoring systems
Leak to the train protection SCADA system	2009	Subway Line - DC, USA	Train collision and deaths
Stuxnet malware attack of nuclear plant	2010	Iran	Increase in centrifuge levels
Backdoor transmission to oil plant	2012	Middle East and North Africa	Data loss, altering the functions
Dragonfly	2013	USA, Spain, France, Italy, German, Turkey, Poland	Email phishing, malware infection by Watering Hole through websites, using SCADA as Trojan
Havex/Energetic Bear RAT	2014	Europe/USA	Closing of hydroelectric dams, overloading power plants
Blackenergy 2/3	2014-2015	Ukraine TV and Energy Sector	Disabling HMI software, disconnection of grids from base station
Crashoverride	2016	Ukraine	Disconnection of grids
Ukraine cyber-attack (Petya malware) (similar to Wannacry)	2017	Ukraine	Disabling of the boards at the airport and metro ticket vending machines, failure of the power plant SCADA connection and some bank services
Triton/Trisis/Hatman	2017	Middle East	Disabling emergency shutdown and security instrument systems

Table 1 presents some critical events and their results on critical infrastructure systems in chronological order. According to a study in 2016, most of the attacks exploited vulnerabilities in level 2 (historian, engineering station, human machine interface..), level 1 components of cell/field zones (RTU, PLC), and the demilitarized zone (DMZ), in that order [10]. Accordingly, vulnerabilities are most commonly observed in devices where supervised control is provided, and then on level 1 devices such as PLC, RTU, and DCS, and then on the manufacturing zone. In addition, vulnerabilities on network devices are observed to be quite high, which leaves a door open to possible attacks using communications infrastructure protocols [10].

The EtherCAT protocol, which is widely used in ICS applications, supports all of the management, cell, field and sensor/actuator levels in computer-integrated manufacturing and meets all communication needs of level 0 to 5 in the Purdue reference model. Due to its wide product range and fast communication features, EtherCAT is used in many sectors, especially in Europe; these sectors include energy, and machine and building automation. The fact that the protocol is Ethernet-based has enabled EtherCAT-based ICS to open up to the outside world with the integration with TCP/IP and many services such as web, FTP and mail which are offered in IT. However, this integration has also made the systems vulnerable to attacks over Ethernet. Furthermore, this protocol does not include authentication, encryption and authorization features like many other ICS protocols; the data is transmitted in plain text and no security mechanism exists at field level. Therefore, the EtherCAT protocol is exposed to IT-based, OT-based, and intrinsic attacks. Thus, it requires a solution to protect against potential cyber threats.

In this study, an EtherCAT-based water level control system testbed was developed, and then the dataset was obtained by generating events. The success and performance of the machine learning methods on EtherCAT-based systems in determining the presence and type of anomalies were evaluated. The main contributions are to develop an EtherCAT-based testbed environment and dataset including attack vectors, and to show the significant effect of the process-based attacks. Since interventions in the ongoing process in ICS are not desirable, most of the anomaly detection studies in the literature apply well-known ICS datasets. In addition, this study evaluates various machine learning methods that are not applied to detect EtherCAT-based network anomalies. There are few ICS studies in the literature based on the EtherCAT protocol. This study contributes to EtherCAT-based ICS environments from a security perspective.

II. RELATED WORK

Before anomalies on the network can be detected, behaviors that are considered normal need to be defined [11]. Accordingly, the normal state of the network is represented by a communications model of the relationships between the fundamental variables, including all system dynamics. An event or object with a certain degree of variation from the formal model is an anomaly. In the literature, anomalies are classified in many ways. Ahmed et al. categorized anomalies as point, contextual and collective [12]. When a particular instance of the flow deviates from the normal event, it is considered to be a point anomaly. A contextual anomaly occurs when an instance behaves anomalously in a particular context. If an event is not an anomaly by itself, but a collection of similar events behaves anomalously, it is defined as a collective anomaly. In addition, Barford and Plonka defined anomalies in three groups: anomalies that are caused by hardware/configuration changes or interruptions in the network; anomalies that occur and disappear over time, such as increased access to a website; and anomalies which derive from manipulation of the network [13]. Similarly, Sestito et al. stated that anomalies can be caused by four factors: attacks, network operations, flash crowds, and measurement errors [14].

In ICS, anomalies can be examined in two main groups: protocol-based and system-based. Protocol-based attacks usually occur through the exploitation of the specifications of the protocol, such as segmentation errors, replay attacks, stack overflows, and fragmentation attacks [15]. System-based attacks can be seen as: database injection attacks [16]; attacks on the PLC RAM where running programs and registers are stored [17]; cryptographic attacks on network devices; and memory overflow or privilege escalation attacks on SCADA software.

In order to detect anomalies on the network, classification, clustering, modeling, statistical, or rule-based techniques are used. Traditional rule-based systems fail with zero-day attacks because too many rules need to be written, and attacks without a signature cannot be detected. Modeling methods are not suitable for intrusion detection of ICS, since a complex model needs to be established. Implementing specification-based solutions is difficult because documents and manuals are inadequate or incomplete on complex ICS [18].

A large number of classification-based methods have been reported in the literature for anomaly detection [18]. The key to ICS anomaly classification is to propose a fast, scalable, and robust solution. For classification, it is in practice difficult to obtain data from the live systems of critical infrastructure. Also, these datasets include attack data, but the data is not well-defined and attacks cannot easily be identified. Thus, the dataset needs to be created in a laboratory environment. In this context, Junejo and Goh developed a water treatment system test environment and generated ten process-based attacks for dataset creation [18]–[24]. Also, Grosso and Sparks proposed a methodology to produce test input for intrusion detection [19], [20]. Maglaras and Yoo developed an attack detection model, but attack vectors were not generated; thus, performance metrics such as FNR (false negative rate) and accuracy could not be evaluated [21], [22]. Antonioli et al. developed honeypots, simulated ICS components, and conducted gamified security competitions in order to create datasets [23]. Ghaeini and Tippenhauer developed SCADA-specific and general network attacks to test the HAMIDS IDS proposal [24].

By integrating TCP/IP-based protocols into critical systems, vulnerabilities in traditional IT networks have been moved to ICS, and this leads to an increase in zero-day attacks [25]. This situation has led researchers to focus more on anomaly-based studies [26], [27]. The anomaly detection studies in IT have a high false-positive (FP) ratio due to the highly variable traffic in the network. However, the cyclic communication of ICS reduces this ratio. Sommer and Paxson stated that machine-learning methods, which are successful in classification, could be used in the detection of attack type rather than intrusion detection, and could also be applied in small and medium networks [28]. It is possible to prevent attacks such as code injection and denial of service (DoS) if the valid commands and the frequency of the commands are known [29]. Gao et al. performed DoS, man-in-the-middle (MITM), and replay attacks on a water level control system developed in the laboratory [30]. Another proposal introduced by Ghaeini et al. is state-aware anomaly detection based on CUSUM computation. The proposal was evaluated on a water SWAT (secure water treatment testbed) [31]. Then, the authors applied anomaly detection based on an artificial neural network (ANN) using a back-propagation algorithm, which takes water level, response frequency and water tank pump on/off state as input. Similarly, Linda et al. generated synthetic attacks using Nmap, Nessus, and Metasploit tools on a control system developed in the laboratory environment and detected the attacks by applying artificial intelligence [32]. These detections were carried out by using the attributes of IP address, interarrival time, number of protocols used, flag code, and total data length; the amount of these data stored was limited to the window size. However, the study was an overview, and semantically insufficient. Ibrahim performed intrusion detection using a supervised ANN model with distributed time-delay neural networks [33]. The study was compared with the detection rates of other ANN-based studies. In another example, Yang et al. performed anomaly detection using autoassociative kernel regression (AAKR) and a statistical probability ratio test (SPRT) in a test environment [34].

It can be seen from the literature that machine learning techniques have not been previously applied to EtherCAT protocol-based ICS for anomaly detection. There are only two studies focusing on anomaly detection of EtherCAT; however, the proposals are rules-based solutions [35], [36]. Furthermore, the EtherCAT protocol also has weaknesses due to the fact that it is Ethernet-based and does not have encryption authentication and authorization mechanisms. With this motivation, in this study, a dataset was created with attack vectors generated on an EtherCAT-based water level control system. Anomaly and attack type detections were evaluated applying various classification techniques.

III. ANALYSIS OF MACHINE LEARNING METHODS IN ETHERCAT-BASED ANOMALY DETECTION

The study comprised: creation of the testbed environment; development of the PLC program for a continuous process on the system; creation of a dataset based on the events generated; determination and reduction of the attributes; and finally, analysis of the machine learning methods for anomaly detection in EtherCAT networks. The goal of the testbed

TABLE 2. Testbed components.

No	Device	Model	Description
1	PS 307 2A	307-1BA00-0AA0	Power supply
2	ET200M	153-1AA03-0XB0	Remote I/O
3	SM 323	323-1BH01-0AA0	DI/DO
4	SM 334	334-0KE00-0AB0	AI/AO
5	KTP600 Basic Color DP	6AV6647-0AC11-3 AX0	HMI panel
6	Micromaster 440	6SE6440-2UC17-5A A1	Drive
7	PC		Engineering station
8	PLC	CX5010	PLC
9	EKxx	EK1110	Extension
10	ETxx	ET2000	Tap
11	ELxx	EL6731	PROFIBUS master terminal

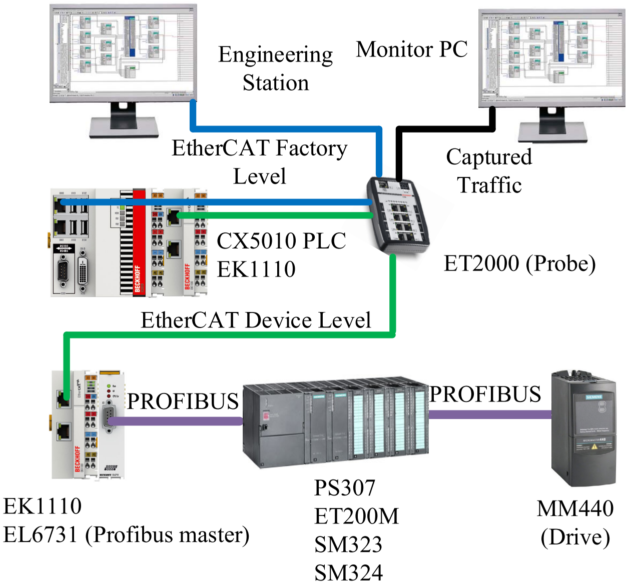

FIGURE 1. Testbed control devices.

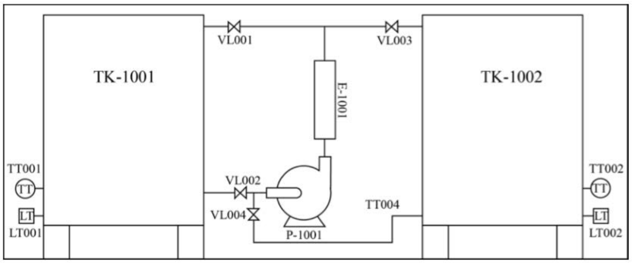

FIGURE 2. TwinCAT diagram.

is to emulate real-world ICS as closely as possible without replicating an entire plant. The anomaly detection part of the study also helps to make those systems more resilient to various security threats.

A. TESTBED

A water level control automation system was created for testing purposes. The first six components in the test environment shown in Table 2 are Siemens devices, and the rest is Beckhoff equipment. The communication was carried out via PROFIBUS and EtherCAT protocols. In order to convert the PROFIBUS system into EtherCAT communication, components 8-11 were added to the system. Fig. 1, which includes field and factory levels and two different communication protocols, and Fig. 2, which shows actuator/sensor level components, complement each other in demonstrating the OT and IT relationship. Although the PLC program of the testbed runs on EtherCAT, the I/O units that perform the job are PROFIBUS-based. Fig. 1 shows the hardware with which the EtherCAT and PROFIBUS protocols communicate. The EL6731 module was used to convert the packets from the EtherCAT protocol to the PROFIBUS protocol, and vice versa. The ET2000 device was used to monitor incoming/outgoing communication packets over a single uplink channel.

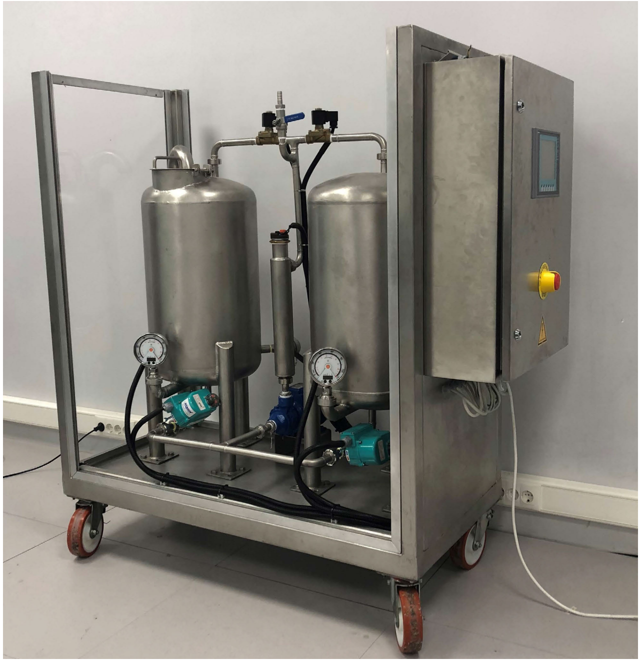

FIGURE 3. Water level control automation components.

PLC programming and all configurations were made by the operator computer with TwinCAT installed (Fig. 1). The CX5010 PLC sent commands to the I/O units of the PROFIBUS system (Fig. 1, Fig. 3) with the help of the EL6731 module. The pump motor was connected to the Micromaster 440 drive. Other I/O units were connected to sensors.

The actuator/sensor-level components controlled by the Micromaster 440 drive and other I/O units are shown in Fig. 2. To control the water level, two tanks (TK-1001/TK-1002), two level sensors (LT001/LT002), two temperature sensors (TT001/TT002), two electric (VL002/VL004) and two solenoid VL001/VL003) valves, one water heater (E-1001), and one pump (P-1001) were used.

B. WATER LEVEL CONTROL AUTOMATION

To control the water level between the tanks, a PLC program was developed and downloaded to the CX5010. Accordingly, when the system was first energized, the water level of each tank was measured by collecting the sensor values. The automation moves to the starting position to flow water from the tank with the larger amount of water to the other. At first, appropriate valves are opened (solenoid and electric valves), then the pump state is changed from ready to operation (start

TABLE 3. Events.

Attack	Group	Category	Description	Intent
A1	G4	Inflow	VL001 is closed	Stop water flow to TK-1001
A2	G4	Inflow	VL003 is closed	Stop water flow to TK-1002
A3	G4	Inflow	VL001 is opened	Start water flow to TK-1001 and TK-1002
A4	G4	Inflow	VL003 is opened	Start water flow to TK-1002 and TK-1002
A5	G4	Outflow	P-1001 is closed	Stop pumping
A6	G4	Outflow	P-1001 is opened	Start pumping
A7	G4	Outflow	VL002 is closed	Stop pumping from TK-1001 to P-1001
A8	G4	Outflow	VL004 is closed	Stop pumping from TK-1002 to P-1001
A9	G3	Tank level	LT001 is changed to 5	Stop process (TK-1001 to TK-1002)
A10	G3	Tank level	LT002 is changed to 5	Stop process (TK-1002 to TK-1001)
A11	G1	System	Link failure	Fault generation
A12	G1	System	Link re-established	Run system after link fault
A13	G4	System	State machine manipulation	Change system status
A14	G2	System	PLC program download and run	Alter running process
A15	G2	System	Replay	Flash-crowd anomaly

command: 0x00207F0C). The pump, which is operated at 2,000 rpm, works until the water level of the tank is at 5%. The pump then goes to the ready state (P-1001 stop: 0x00007E04) and the valves (VL002-VL003 or VL001-VL004) are closed. The system stands by for about 15 seconds and then starts to move in the opposite direction to the previous operation. Thus, a cycle is completed and a continuous process is created for testing purposes. Scaling is carried out for water-level information gathered from sensors, and the data obtained was converted to the range 0-100 (LO_LIM, HI_LIM) using the following formula:

$$\begin{aligned} \mathrm{OUT} = [((\mathrm{FLOAT(IN)} - \mathrm{K1})/(\mathrm{K2} - \mathrm{K1})) * (\mathrm{HI\_LIM} \\ - \mathrm{LO\_LIM})] + \mathrm{LO\_LIM} \end{aligned}$$

The sensors send input values in a unipolar format. Therefore, the input values obtained ranged from 0 to 27648 such that K1 = 0.0 and K2 = +27648.0. In addition, the data obtained from the PROFIBUS environment, excluding driver data, was converted into a little-endian format, which is used by TwinCAT PLC.

C. EVENT GENERATION

Using the testbed, a dataset was created from the system which was later used for intrusion detection. To generate the dataset, 16 events were created, including 15 different attacks and normal system behavior (Table 3). In Table 3, attacks are formed into four different groups:

-G1: Events that occur in the network due to hardware, configuration changes or interruptions.

-G2: Anomalies that have arisen over time due to agglomeration, such as a sudden increase in access to a software-based system or website but calm down over time.

-G3: Anomalies caused by measurement errors.

-G4: Abuse of network.

These groups of attacks may affect four different structures: the system, tank level, and incoming and outgoing flows. To obtain normal network traffic, the system was run for one hour and the event is represented as A0. For state machine manipulation, OP status was changed respectively to INIT, PREOP, SAFEOP, and to OP status again. For a link failure event, connection to I/O units was interrupted, and then connection was reactivated. In addition, for A14, a PLC program which runs the motor connected to the drive was developed, and the system behavior was captured during the program download to the PLC. To monitor the system, the ET2000 probe device was placed between the EL6731 module and the PLC.

There are 14 different types of attack and unexpected event. Small events such as ''link failure'' could have a greater meaning when combined with other events. For instance, advanced persistent threat (APT) attacks are targeted attacks: they start with the discovery of the topology and try to copy themselves to the nodes in the network, which have a connection to the engineering station. Then, the complex part of the attack starts. Thus, even APT attacks begin with simple steps for activation. APT detection solutions try to detect both simple and complex events within the same solution. Our goal is to detect as many events as we can (attacks and other possible events occurring in the use case) and present a holistic view of the detected attacks to the user to assist in further influencing the outcome.

D. STATISTICAL IDENTIFIERS, ATTRIBUTE SELECTION AND REDUCTION

A three-step process was applied to identify and reduce the attributes in the dataset.

1) WIRESHARK POST-DISSECTOR DEVELOPMENT

The dissector written in Lua converts all events to the appropriate format for further processing. When this extension was activated, commands, padding data, data lengths, and registers used in the PCAP records were parsed and presented in the top panel. Thus, statistical values were identified for each packet (Table 4).

2) ATTRIBUTE DETERMINATION

The packets were exported in.csv format over Wireshark and input to a program developed for further parsing. The program took the packets' attributes given in Table 4 and evaluated the total and average values in a predefined window. Window size was taken as one second. Selecting a small window size makes it difficult to capture attacks, while select-

TABLE 4. Attributes used in the analysis.

Attr.	Description
1	Total number of packets in window
2	Mean of packets in window
3	Sum of padding data sizes of packets in window
4	Average of padding data sizes of packets in window
5	Throughput in window (byte/sec)
6	Average throughput in window
7	Number of different master source addresses in window
8	Different master destination addresses in window
9	Average of Nope command data size in window
10	Sum of Nope command data size in window
11	Average of APRD command offset (register) values in window
12	Sum of APRD command offset values in window
13	Average of APWR command offset (register) values in window
14	Sum of APWR command offset (register) values in window
15	Data size average of APRW command in window
16	Data size summation of APRW command in window
17	Average of FPRD command offset (register) values in window
18	Sum of FPRD command offset (register) values in window
19	Average of FPWR command offset (register) values in window
20	Sum of FPWR command offset (register) values in window
21	Data size average of FPRW command in window
22	Data size summation of FPRW command in window
23	Average of BRD command offset (register) values in window
24	Sum of BRD command offset (register) values in window
25	Average of BWR command offset (register) values in window
26	Sum of BWR command offset (register) values in window
27	Data size average of BRW command in window
28	Data size summation of BRW command in window
29	Data size average of LRD command in window
30	Data size summation of LRD command in window
31	Data size average of LWR command in window
32	Data size summation of LWR command in window
33	Data size average of LRW command in window
34	Data size summation of LRW command in window
35	Average of ARMW command offset (register) values in window
36	Sum of ARMW command offset (register) values in window
37	Data size average of FRMW command in window
38	Data size summation of LWR command in window
39	The average of the numeric value of the NOP command data field in window
40	The sum of the numeric value of the NOP command data field in window
41	The average of the numeric value of the LRD command data field in window
42	The sum of the numeric value of the LRD command data field in window
43	The average of the numeric value of the LWR command data field in window
44	The sum of the numeric value of the LWR command data field in window
45	The average of the numeric value of the APRD command data field in window
46	The sum of the numeric value of the APRD command data field in window
47	The average of the numeric value of the LRW command data field in window
48	The sum of the numeric value of the LRW command data field in window
49	The average of the numeric value of the APRW command data field in window
50	The sum of the numeric value of the APRW command data field in window
51	The average of the numeric value of the FPRW command data field in window
52	The sum of the numeric value of the FPRW command data field in window
53	The average of the numeric value of the BRW command data field in window
54	The sum of the numeric value of the BRW command data field in window

ing a large window size will cause packets in the previous or next cycle to be added to the present cycle. In addition, a large window size decreases the impact of unit statistics on frames containing attacks. The descriptions of the EtherCAT commands in Table 4 are given in Table A5, Appendix. More details on standard EtherCAT protocol commands can be found in [36].

3) ATTRIBUTE SELECTION

In order to determine the effect of the attributes on the attack vectors and normal behavior for each event, dominant attributes were determined and unnecessary ones eliminated. Most of the attributes are formed by EtherCAT protocol commands as they are used for read/write data. For the selection, attribute size was reduced by using regression analysis.

Descriptive statistics in the dataset for the one-second window size are given in [Table A1, Appendix]. The statistics are mostly EtherCAT commands and some other properties such as summation, average, group, or attack type. In the statistics, it can be seen that APRW command (Attributes 15, 16), FPRW command (Attributes 21, 22), BRW command (Attributes 27, 28), FRMW command (Attributes 37, 38), LWR command data (Attributes 43, 44), APRD command data (Attributes 45, 46), APRW command data (Attributes 49, 50), FPRW command data (Attributes 51, 52), and BRW command data (Attributes 53, 54) contain no information. Thus, the attributes were initially reduced to 32 by removing relevant attributes from the dataset. After this stage, to select the remaining attributes, a regression equation was applied using two different approaches: single event-based, and all events-based representative dataset. In this way, the significance of each attribute could be evaluated from both event-based and total dataset perspectives. In selecting an attribute, the significance of a variable was determined by the p-value [37]–[39].

1. **Single event-based dataset**: Sub-datasets were prepared for each attack situation. Each dataset had only attack or non-attack status. Accordingly, appropriate variables were found for each attack [Table A2, Appendix]. The most significant variables based on the attacks are Sum LRW data, Sum ARMW command, Sum APRD command, Sum APWR command, Avg LRW data, Sum FPRD command, and SumPacketLength attributes.

2. **All events-based sample dataset**: A sample dataset containing all attack and normal behaviors was randomly formed from the main dataset. Significant variables were determined by a regression equation. In the sub-dataset, 180 data were studied with the most significant 32 attributes. The attributes were reduced to 11 by removing insignificant ones using regression. Then, the regression equation was reconstructed by adding the attributes that were significant in the first step to the reduced 11 attributes. The overlapping and insignificant attributes were removed from the equation. The regression resulted in a new model with 10 attributes, which was derived from previous models and is shown in Table 5. At the end of the regression equation, SumPadByte, SumPacketLength, Sum FPWR command, Sum BRD command, Sum LRD command, Sum LRD command data, Sum LRW command data, Avg NOP command data, Avg LRD command data and Avg LRW command data attributes were found as significant in determining the attack and normal behavior events. The correlation coefficient of the equation is 0.837. This finding shows that the relationship between the attributes and the attacks is strong. Similarly, the significance level of the equation was found to be 1.2×10^{-19}. This shows that the level of significance is very close to zero, and the equation is statistically significant.

When single event-based and all event-based sample datasets are compared, Sum LRW attribute is found to be critical. In all models, this variable has a high level of significance ($P<0.1$). In addition, SumPadByte, SumPacketLength, Sum FPWR and Sum BRD attributes are found to be significant in both cases.

TABLE 5. Regression equation for sample dataset and significance of attributes.

	Coefficients	Standard Error	t Stat	P-value
Intercept	23.97471	2.924345	8.198316	6.00E-14
SumPadByte	-0.00887	0.002315	-3.83273	0.000179
SumPacketLength	0.005112	0.00133	3.842959	0.000172
Sum FPWR command	-0.00017	9.35E-05	-1.83161	0.06878
Sum BRD command	-0.00912	0.002433	-3.75025	0.000243
Sum LRD command	0.044081	0.01264	3.48731	0.000623
Sum LRD data	64.27536	21.23337	3.027091	0.002858
Sum LRW data	0.006807	0.001494	4.556349	9.97E-06
Avg NOP data	33.16901	9.235732	3.591379	0.000432
Avg LRD data	-107,208	35206.69	-3.04509	0.002701
Avg LRW data	-0.66086	0.115392	-5.72704	4.61E-08

TABLE 6. Dataset distribution.

	Total Attack Types	Attack	Normal	Total
Training	15	94	3653	3744
Test	15	43	1565	1604

IV. ATTACK DETECTION ON THE WATER-LEVEL CONTROL AUTOMATION

This section describes how attack detection was performed using the attributes obtained from the regression analysis. The results of the 15 different attacks are also presented. A total of 5348 data (5348 seconds) was divided into two parts: 70% training and 30% test which is a commonly accepted distribution in the literature. The training dataset consisted of 3744 data, where 94 were attack and 3653 were normal network traffic data. On the test side, the dataset comprised 43 attack and 1565 normal behavior data. The random sampling method was chosen for the separation of the training and test datasets, but the attack data was divided into two, since each attack should be present in both the training and test datasets. For example, if an attack has 15 data, 11 are included in the training dataset, and four are included in the test dataset. Because the number of data of some attacks is very low, the same attack exists in both datasets. The summary of the data is shown in Table 6.

Using the dataset, 18 different techniques were applied for anomaly detection by using the RapidMiner, Weka, MATLAB and Excel programs for the models [Table A3, Appendix]. Among the techniques used, the worst results were obtained with the classification by polyregression (accuracy 0.11%, recall 0%). The results of four techniques were salient and gave better results than others; these are ANN, decision trees, support vector machine with genetic algorithm (SVM GA), and k-nearest neighbor (k-NN).

A back-propagation algorithm was used in ANN, and the training cycle was 500 iterations. The learning coefficient was 0.3 and the moment value was 0.2. The target error value was set to 0.00001, and the entire dataset was normalized in the range $[-1\ 1]$. ANN was applied with a single layer using 80 neurons and a sigmoid type of activation function was selected.

In the decision trees, the decision to branch the tree was determined according to the gain ratio. Since the maximum depth of the tree could be memorized if it was unlimited, it was decided to set it low enough not to be memorized but large enough to detect the attacks; the value was determined to be 20. In order to prevent expansion in the tree, pruning is also important. The pruning limit was set at 0.25 and an optimistic model created. The lowest gain was taken as 0.1. This value was calculated before the node was split; if the gain was less than this value, the node branching was done by division. After fragmentation, the minimum data size of each leaf was taken as four. This is because attacks range from 1 to 32 data. If this value is high, the attacks cannot be determined. If it is very low (e.g., 1), the tree branches out more.

Another technique used was SVM GA. For SVM, the radial kernel was selected. The radial core was defined as $e^{(-g||x-y||2)}$. Here g indicates the gamma value. The corrected gamma parameter is the most important parameter in the core performance and needs to be carefully determined depending on the problem. On the GA side, the initial population was randomly assigned and a maximum of 10 000 generations produced. If there was no improvement in more than 30 generations, the algorithm stopped as there could not be any progress. GA selection was made by the tournament method and the fraction value was 0.75. GA mutation type was Gaussian. The possibility of crossing was taken as 1.

The last technique, k-NN, is simpler than the other three methods and is among the most basic. Two attributes take the form of a plane and three attributes take the form of a volume structure, while the four and above attributes have no geometric representation. Here, as the number of attributes increased, more accurate results were obtained. In k-NN, the grouping was calculated by Euclidean distance using the attributes.

V. RESULTS AND CONCLUSION

The estimation of whether an attack exists or not is binomial behavior. Therefore, predictions can be shown as a binary classification. In binary estimation, classification is categorized into accuracy, precision, and recall. For binary classification, these approaches are determined using the sample confusion matrix. There are four cases in the matrix:

- True Positives (TP): Actual attack, predicted as attack

- True Negatives (TN): Actual non-attack (normal network traffic) and predicted as non-attack

- False Positives (FP): Actual non-attack, predicted as attack, (also known as Type I error)

- False Negatives (FN): Actual attack, predicted as non-attack (also known as Type II error) On this basis:

TABLE 7. Prediction results of models.

	Accuracy	Recall	Precision
Artificial Neural Networks (ANN)	98.28%	100.00%	98.27%
Decision Trees (DT)	98.17%	100.00%	98.16%
SVM GA	99.81%	100.00%	99.81%
k-NN	100.00%	100.00%	100.00%

- Accuracy = TP + TN (true estimations in both classes) / all data

- Sensitivity = (TP) / (TP + FP)

- Recall = TP / (TP + FN)

Accordingly, the results of the four models are shown in Table 7. It can be said that decision trees give the lowest performance among the four techniques. The reason is that the number of data per leaf should be at least four in the tree structure. This rule classifies network data rows with similar attributes. In other words, some of the four rows of data may be attack, some may not be attack, and this resulted in lower value in terms of accuracy and precision than others. Another important part of the tree structure was the tree depth limit, which was 20. This limit prevented the tree from having too much depth, and thus prevented the fragmentation of data and the interpretation of smaller data. However, it should be noted that the accuracy of 98.17Compared to the other three models, it has an accuracy of approximately 1.6and SVM GA give better results, respectively. The results for ANN are close to those of decision trees but more reasonable. The use of a single hidden layer as a network model and the model setup with 80 neurons led to lower learning accuracy compared to SVM GA and k-NN. The fact that the training iteration value was 500 may also have caused the process to end before the training was completed but considering the number of neurons, it seems that 500 iterations are suitable for training. In the SVM GA model, the results were better than the other two models because of the 10 000 generations. Thus, the GA solution cluster begins to optimize over a wider range; it optimized the solution cluster range in less time and spent more effort while finding a lower error. The k-NN model gave the best predictions and correctly determined all of the attacks. The reason is that it can reach as many dimensions as the number of attributes. In k-NN, for each sample value, the single nearest value is examined. The sample is included in the class of the closest value (attack or normal traffic). If the two nearest samples were selected, they would not be able to perform the correct grouping when the two samples were from different classes. Therefore, the closest single sample selection is more appropriate, thus ensuring high accuracy, recall and precision values. Furthermore, with the large number of dimensions, the attack states could be divided into areas of different dimensions; thus, all attacks and normal traffic could be identified.

The prediction results showed that the attacks could be flagged in four models with high accuracy. For ICS networks, it is also important to determine the attack type. As a further output of the study, the results of the attack type predictions are shown in Table 8 and Fig. 4.

TABLE 8. Attack type prediction results of models.

	A0	A1	A2	A3	A4	A5
Decision Trees	100.00%	0.00%	0.00%	0.00%	0.00%	0.00%
k-NN	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
ANN	100.00%	0.00%	0.00%	0.00%	0.00%	0.00%
SVM GA	100.00%	0.00%	100.00%	46.15%	50.00%	50.00%
	A6	**A7**	**A8**	**A9**	**A10**	**A11**
Decision Trees	0.00%	0.00%	0.00%	0.00%	0.00%	100.00%
k-NN	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
ANN	0.00%	0.00%	0.00%	0.00%	0.00%	100.00%
SVM GA	0.00%	22.22%	57.14%	34.38%	85.71%	85.71%
	A12	**A13**	**A14**	**A15**		
Decision Trees	0.00%	92.86%	0.00%	100.00%		
k-NN	100.00%	100.00%	100.00%	100.00%		
ANN	0.00%	100.00%	57.14%	100.00%		
SVM GA	0.00%	35.71%	57.14%	83.33%		

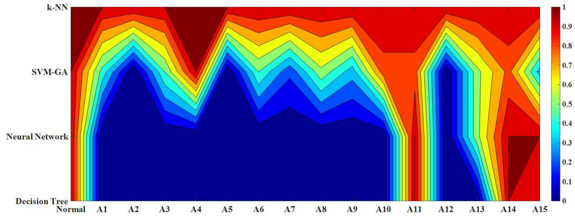

FIGURE 4. Surface graphic of estimation results by attack type.

All four models correctly predicted normal network traffic. The A1 attack was correctly identified by only the k-NN model. The important point here is that the A1 attack has only one data which makes it difficult for models to predict. Although there is one data in the A2 attack as well, both the k-NN and SVM GA models made accurate predictions. It can be seen that the attribute Sum LRW (Data5) is important in the A2 attack type [Table A2, Appendix]. In the A3 attack, SVM GA found half of the attacks, while the remaining attacks were grouped as normal network traffic. Again, for the A3 attack, k-NN was able to identify all the attacks. ANN and decision trees could not determine A4, A5, A6, A7, A8, A9, A10, and A12 attacks correctly. The reason for this is that the diffraction size of the classifiers is not good enough in models other than k-NN and SVM GA. For the A4 attack, the SVM GA model predicted half of the attacks correctly, while the remaining 80% of attacks were classified as A3, and 20% were classified as A1. In the A5 attack, SVM GA determined half of the attacks correctly. However, in some cases, it classified the same attack as A1, A4, and A6. For A6, SVM GA determined the event as an attack but the method could not classify the attack type correctly (classified as A1 and A2). The situation with A7 was similar, where 22.22% of the data was correctly grouped and the remaining 77.78% of the data classified as normal network traffic, A2, A5, and A6 attacks. 57.14% of the A8 attack was classified correctly and the rest was identified as A6 or A7. In A9 attacks, 34.38% were correct estimations, 34.3817% were A7, 8.5% were A6, and the remaining estimations were A4. In fact, the method found the presence of attack correctly but could not determine the exact type. 14.29the A10 attack was classified as A8 by SVM GA. The worst performance on the A11 attack classification was obtained with SVM GA. While ANN, decision trees and the k-NN model found all

A11 attacks, SVM GA identified 14.29 of these attacks as A10 attacks. In the A12 attack, SVM GA determined attack type as A8, though it did flag the event as an attack. ANN and k-NN correctly identified all attacks of A13. Decision trees determined 7.14% of A13 attack data as normal network traffic, whereas SVM GA classified 57.14% as A15 and 7.14% as A7 attack type. While the decision trees identified the A14 attack as normal network traffic, ANN determined 14.28% of the attack as A12 and 28.56% as normal network traffic. The SVM GA model grouped 14.28% of A14 attacks as A11 and 28.56% as A12. Similarly, 8.33% and 8.33% of A15 were classified as A11 and A13.

In general, the most accurately predicted events in intrusion detection models are normal network traffic, A11, A15 and A13, respectively. Then, the prediction performance is A2, A10, A8, A4, A5, A3, A9, and A7 events, respectively. The models showed the worst performance in A1, A6, and A12 attacks. The main reason for poor prediction is that the number of data to be estimated was very low in these events.

Fig. 4 shows the thermal graph presenting the rate of correct prediction of the attack type according to the techniques. The red areas show the accuracy of the grouping estimations and the dark blue areas show the false grouping. Accordingly, it is clear that the k-NN and SVM GA models perform better than ANN and decision trees. ANN and decision trees, especially in the A1 to A10 range and the A12 attacks, performed poorly in determining the attack group.

Results show that accuracy, precision, and recall rates are acceptable when compared to previous studies; [14] and [18] are the most similar studies in terms of applied algorithms and testbed environment. In [14], accuracy was found to be between 92% and 99% for various ANN models, whereas the highest accuracy was found to be 100% with k-NN in our work. In [18], anomalies on the water management testbed were found by applying algorithms such as Best-first tree (BFTree), ANN and SVM. The best results were obtained by BFTree with a detection rate of 99.72%. In our study, while k-NN has the best accuracy of 100%, the decision trees method was applied and 98.17% accuracy was achieved. In [18], performance values are found to be 98.24% accuracy, 98.40% precision, and 98.20% recall rates with ANN, whereas our study has 98.28%, 100%, and 98.27%, respectively. SVM performance is also similar, with98.71% accuracy, 98% precision and 98.7% recall in [18], compared with 99.81% accuracy, 100% precision and 99.81% recall in our work. The techniques and results used in the comparative studies are in line with our study. As in other studies, the testbed environment was developed with the physical and control components of a real process. Unlike other studies, EtherCAT was used as the communication protocol. We conclude that anomaly detection can be performed with machine learning techniques in EtherCAT protocol-based systems, which is one of the critical infrastructure systems where it is difficult to obtain datasets by their nature.

In [36], the trust node communication approach was proposed; this relies on detecting intrusions coming from unapproved nodes in the network. However, there might be anomalies sourced from the trust nodes as well. This study fills the gap in anomalies based on approved nodes and thus the anomaly detection is considered holistically.

VI. FUTURE WORK

Future work, the most successful model among the applied methods can be integrated into a previously developed EtherCAT-based IDS system [36], which will determine anomalies in real-time packets. Another possible area of work is to improve the proposal to detect possible malicious activities on acyclic EtherCAT traffic. The challenge in this work is that the variability of the pattern in the acyclic ICS factory-level communication is greater than it is in field communication.

ACKNOWLEDGMENT

The authors would like to thank Mustafa Akpinar for his helpful discussions and valuable suggestions.

REFERENCES

[1] L. Obregon, "Secure architecture for industrial control systems," SANS Inst. Inf. Secur. Reading Room, Bethesda, MD, USA, Sep. 2015.

[2] T. J. Williams, "A reference model for computer integrated manufacturing from the viewpoint of industrial automation," *IFAC Proc.*, vol. 23, no. 8, pp. 281–291, Aug. 1990.

[3] K. Stouffer, V. Pillitteri, S. Lightman, M. Abrams, and A. Hahn, "Guide to industrial control systems (ICS) security," Nat. Inst. Standards Technol., Gaithersburg, MD, USA, Jun. 2015.

[4] P. Kevin, *Slammer Worm Crashed Ohio Nuke Plant*. SecurityFocus. Accessed: Apr. 8, 2016. [Online]. Available: http://www.securityfocus.com/news/6767

[5] S. Tony. 2001. *Hacker Jailed for Revenge Sewage Attacks*. Accessed: Apr. 8, 2016. [Online]. Available: http://www.theregister.co.uk/2001/10/31/hacker_jailed_for_revenge_sewage/

[6] M. Abrams and J. Weiss, "Control system cyber security case study," Nat. Inst. Standards Technol., Bellingham, WA, USA, 2007.

[7] "Safety study: Supervisory control and data acquisition (SCADA) in liquid pipelines," Nat. Transp. Saf. Board, Washington, DC, USA, Tech. Rep. PB2005-917005, 2005.

[8] J. Weiss, "A review of selected actual control system cyber incidents," in *Proc. Fall Conf. (ICSJWG)*, 2009.

[9] B. Miller and D. Rowe, "A survey SCADA of and critical infrastructure incidents," in *Proc. Annu. Conf. Res. Inf. Technol.*, 2012, pp. 51–56.

[10] Fireeye Isight Intelligence, "Overload critical lessons from 15 years of ICS vulnerabilities," ICS Vulnerabılıty, CA, USA, Trend Rep., 2016.

[11] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Network anomaly detection: Methods, systems and tools," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 303–336, 1st Quart., 2014.

[12] M. Ahmed, A. N. Mahmood, and J. Hu, "A survey of network anomaly detection techniques," *J. Netw. Comput. Appl.*, vol. 60, pp. 19–31, Jan. 2016.

[13] P. Barford and D. Plonka, "Characteristics of network traffic flow anomalies," in *Proc. 1st ACM SIGCOMM Workshop Internet Meas. (IMW)*, 2001, p. 69.

[14] G. S. Sestito, "A method for anomalies detection in real-time Ethernet data traffic applied to PROFINET," *IEEE Trans. Ind. Informat.*, vol. 14, no. 5, pp. 2171–2180, May 2018.

[15] N. Sayegh, A. Chehab, I. H. Elhajj, and A. Kayssi, "Internal security attacks on SCADA systems," in *Proc. 3rd Int. Conf. Commun. Inf. Technol. (ICCIT)*, 2013, pp. 22–27.

[16] B. Zhu, A. Joseph, and S. Sastry, "A taxonomy of Cyber attacks on SCADA systems," in *Proc. Int. Conf. Internet Things 4th Int. Conf. Cyber, Phys. Social Comput.*, 2011, pp. 380–388.

[17] R. E. Johnson, "Survey of SCADA security challenges and potential attack vectors," in *Proc. Int. Conf. Internet Technol. Secured Trans.*, 2010, p. 5.

[18] K. N. Junejo and J. Goh, "Behaviour-based attack detection and classification in cyber physical systems using machine learning," in *Proc. 2nd ACM Int. Workshop Cyber-Phys. Syst. Secur. (CPSS)*, 2016, pp. 34–43.

[19] C. Del Grosso, G. Antoniol, E. Merlo, and P. Galinier, "Detecting buffer overflow via automatic test input data generation," *Comput. Oper. Res.*, vol. 35, no. 10, pp. 3125–3143, Oct. 2008.

[20] S. Sparks, S. Embleton, R. Cunningham, and C. Zou, "Automated vulnerability analysis: Leveraging control flow for evolutionary input crafting," in *Proc. 23rd Annu. Comput. Secur. Appl. Conf. (ACSAC)*, 2007, pp. 477–486.

[21] L. A. Maglaras and J. Jiang, "Intrusion detection in SCADA systems using machine learning techniques," in *Proc. Sci. Inf. Conf.*, 2014, pp. 626–631.

[22] S. Lee, H. Yoo, J. Seo, and T. Shon, "Packet diversity-based anomaly detection system with OCSVM and representative model," in *Proc. IEEE Int. Conf. Internet Things (iThings) IEEE Green Comput. Commun. (GreenCom) IEEE Cyber, Phys. Social Comput. (CPSCom) IEEE Smart Data (SmartData)*, Dec. 2016, pp. 498–503.

[23] D. Antonioli, H. R. Ghaeini, S. Adepu, M. Ochoa, and N. O. Tippenhauer, "Gamifying ICS security training and research," in *Proc. Workshop Cyber-Phys. Syst. Secur. PrivaCy (CPS)*, 2017, pp. 93–102.

[24] H. R. Ghaeini and N. O. Tippenhauer, "HAMIDS: Hierarchical monitoring intrusion detection system for industrial control systems," in *Proc. 2nd ACM Workshop Cyber-Phys. Syst. Secur. Privacy (CPS-SPC)*, 2016, pp. 103–111.

[25] Y. Yang, K. McLaughlin, T. Littler, S. Sezer, B. Pranggono, and H. F. Wang, "Intrusion detection system for IEC 60870-5-104 based SCADA networks," in *Proc. IEEE Power Energy Soc. Gen. Meeting*, Jul. 2013, pp. 1–5.

[26] M. Cheminod, L. Durante, and A. Valenzano, "Review of security issues in industrial networks," *IEEE Trans. Ind. Informat.*, vol. 9, no. 1, pp. 277–293, Feb. 2013.

[27] G. N. Ericsson, "Cyber security and power system communication-essential parts of a smart grid infrastructure," *IEEE Trans. Power Del.*, vol. 25, no. 3, pp. 1501–1507, Jul. 2010.

[28] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," in *Proc. IEEE Symp. Secur. Privacy*, May 2010, pp. 305–316.

[29] R. R. R. Barbosa, R. Sadre, and A. Pras, "Exploiting traffic periodicity in industrial control networks," *Int. J. Crit. Infrastruct. Protection*, vol. 13, pp. 52–62, Jun. 2016.

[30] W. Gao, T. Morris, B. Reaves, and D. Richey, "On SCADA control system command and response injection and intrusion detection," in *Proc. eCrime Res. Summit*, 2010, pp. 1–9.

[31] H. R. Ghaeini, D. Antonioli, F. Brasser, A.-R. Sadeghi, and N. O. Tippenhauer, "State-aware anomaly detection for industrial control systems," in *Proc. 33rd Annu. ACM Symp. Appl. Comput. (SAC)*, 2018, pp. 1620–1628.

[32] O. Linda, T. Vollmer, and M. Manic, "Neural network based intrusion detection system for critical infrastructures," in *Proc. Int. Joint Conf. Neural Netw.*, 2009, pp. 1827–1834.

[33] L. M. Ibrahim, "Anomaly network intrusion detection system based on distributed time-delay neural network," *J. Eng. Sci. Technol.*, vol. 5, no. 4, pp. 457–471, 2010.

[34] D. Yang, A. Usynin, and W. J. Hines, "Anomaly-based intrusion detection for SCADA systems," in *Proc. 5th Int. Top. Meeting Nucl. Plant Instrum., Control Hum. Mach. Interface Technol. (NPIC&HMIT)*, 2006.

[35] A. Granat, H. Höfken, and M. Schuba, "Intrusion detection of the ICS protocol EtherCAT," in *Proc. 2nd Int. Conf. Comput., Netw. Secur. Commun. Eng.*, 2017, pp. 113–117.

[36] K. O. Akpinar and I. Ozcelik, "Development of the ECAT preprocessor with the trust communication approach," *Secur. Commun. Netw.*, vol. 2018, Apr. 2018, Art. no. 2639750, doi: 10.1155/2018/2639750.

[37] M. Stuart and S. M. Ross, "Introduction to probability and statistics for engineers and scientists," *J. Roy. Stat. Soc. A (Statist. Soc.)*, 2006.

[38] V. M. Abraham, R. E. Walpole, and R. H. Myers, "Probability and statistics for engineers and scientists," *Math. Gazette*, 2007.

[39] B. C. Gupta and I. Guttman, *Statistics and Probability With Applications for Engineers and Scientists*, 1st ed. Hoboken, NJ, USA: Wiley, 2013.

KEVSER OVAZ AKPINAR (M'18) was born in Turkey, in 1986. She received the B.Sc. degree in computer engineering from Pamukkale University, Turkey, the M.Sc. degree in computer science with a major in network security from The University of Texas at San Antonio, USA, and the Ph.D. degree in computer engineering with a major in industrial control systems (ICS) security from Sakarya University, Turkey. Her research interests include big data, vehicular communication, SCADA security, and ICS protocols.

IBRAHIM OZCELIK was born in Turkey, in 1973. He received the B.Sc. degree in electric-electronical engineering from Karadeniz Technical University, Turkey, and the M.Sc. degree in electronic engineering with a major in Traffic Control Mechanism in ATM Network and the Ph.D. degree in electronic engineering with a major in CAN/ATM and Profibus/ATM local bridge development from the Sakarya University, Turkey, in 2002. His main research interests are industrial networks, SCADA protocols, VANETs, wireless networks, and cyber security issues.