IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 14, 2019, accepted July 1, 2019, date of publication July 5, 2019, date of current version July 25, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2927037

Protecting Cyber Physical Systems Using a Learned MAPE-K Model

IBRAHIM ELGENDI[1], (Member, IEEE), MD. FARHAD HOSSAIN[2], (Member, IEEE), ABBAS JAMALIPOUR[3], (Fellow, IEEE), AND KUMUDU S. MUNASINGHE[1], (Member, IEEE)

[1]Faculty of Education, Science, Technology and Mathematics, University of Canberra, Canberra, ACT 2601, Australia
[2]Department of Electrical and Electronic Engineering, Bangladesh University of Engineering and Technology, Dhaka 1216, Bangladesh
[3]School of Electrical and Information Engineering, The University of Sydney, Sydney, NSW 2006, Australia

Corresponding author: Ibrahim Elgendi (Ibrahim.Elgendi@canberra.edu.au)

ABSTRACT Industry 4.0 leverages on cyber-physical systems (CPSs) that enable different physical sensors, actuators, and controllers to be interconnected via switches and cloud computing servers, forming complex online systems. Protecting these against advanced cyber threats is a primary concern for future application. Cyberattackers can impair such systems by producing different types of cyber threats, ranging from network attacks to CPS controller attacks, which could impose catastrophic damage to CPS infrastructure, companies, governments, and even the general public. This paper proposes a learned monitor, analyze, plan, execute, and knowledge (MAPE-K) base model as a method for supporting self-adaptation for the CPSs, ensuring reliability, flexibility, and protection against cyber threats. The model aims to gauge normal behavior in an industry environment and generate alarms to alert users to any abnormalities or threats. In turn, our evaluation shows 99.55% accuracy in detecting cyber threats.

INDEX TERMS Legitimate, malicious attacker, monitor, analysis, planning, execution, and knowledge base model, machine learning.

I. INTRODUCTION

Cyber-physical systems (CPS) have become the backbone of modern automation and data exchange in manufacturing technologies, more commonly known as Industry 4.0. This includes CPSs such as the Internet of Things (IoT), cloud computing, and cognitive computing. Therefore, protecting such systems against different types of cyber threats has become a challenge in the modern industry, wherein data protection, safety, and security are top priorities [1]. Reliability and availability of CPS communications, which can be considered as highly heterogeneous and dynamic, are affected by numerous threats. Essentially, these communications provide interconnections between sensors, actuators, and controllers. Within these environments, controllers connect to switches, which eventually forge links to computing servers. In this sense, heterogeneous communication networks can be viewed as the main enabler between operational technology and information technology (IT) in the next phase of the industrial revolution.

The associate editor coordinating the review of this manuscript and approving it for publication was Zonghua Gu.

Moreover, modern industrial systems have evolved into autonomous IoT and CPS systems that require greater reliability and availability [2]; however, this has made Industry 4.0 vulnerable to cyberattacks. As a result, the Industry Internet Consortium [3] identified a broad range of possible weaknesses that may threaten this revolution. For example, smart sensors, actuators, controllers combined with programmable logic control (PLC), and servers combined with control software are just some potential vulnerabilities. In addition, the National Institute of Standards and Technology (NIST) deemed a range of influential factors possible cyber threats to industry environments, including PLC manipulation, denial of control actions, and spoofed computing servers [4]. Cyber threats could also be exposed to different system layers [5]. Notably, sensor and actuator tiers may be subject to brute force attacks, while the network tier may be subject to flooding transmission control protocol (TCP) SYN attacks (a type of denial-of-service [DoS] attack). Similarly, the control tier may be threatened by remote users changing PLC parameters.

Current networking technologies provide great capability to detect different types of cyber threats, ranging from DoS attacks affecting networks, to PLC attacks affecting CPS controllers. Notably, recent data link technologies have

led to software running over networks in Industry 4.0 systems, making them relatively more flexible, powerful, and operationally simpler. SYN flooding attacks exhaust network resources, starting from the data link layer, computing server, memory, router, and finally to end-host resources to exhaust the backlog of half-open connections corresponding to each port number for any application in an industrial environment [6]. The aim is to send a quick spoofed SYN segment to the server using a spoof IP address, and to refrain from responding to SYN + ACK segments produced by the server [7]–[9].

The monitor, analyze, plan, execute, and knowledge (MAPE-K) model in [10] was used to enable Industry 4.0 to self-adapt, self-heal, self-optimize, self-secure, and gain self-control [11], [12]. This existing model relies on predefined self-adaptation policies such as event–condition–action (ECA) policies, goal policies, or utility function policies [13], which are generated using automated planning. Importantly, the MAPE-K cannot detect new cyber threats if they are not predefined in ECA; thus, this represents a major problem for Industry 4.0.

Therefore, this paper proposes a new control system architecture and learned MAPE-K model that can gauge and detect any cyber threats targeted at CPSs in Industry 4.0. As a result, such systems will be able to self-predict any threats and protect themselves against unlawful breaches of data and/or control parameters, including spoofed SYN flooding attacks.

This paper contributes to the literature by:

- developing a new control system architecture for Industry 4.0
- developing a learned MAPE-K model based on a support vector machine–radial basis function (SVM–RBF) machine learning (ML) method to gauge normal behavior and generate alerts for abnormalities in Industry 4.0 environments.

The remainder of this paper is organized as follows. Section II provides the related work, while Section III explores Industry 4.0 system architecture. Sections IV and V next present the evaluation and conclusion, respectively.

II. RELATED WORKS

As CPSs become more intelligent, interconnected, and coupled with physical devices, security grows in concern. Security-related information is required to build different activities ranging from security analysis to design security control. The scientific community proposed security approaches based on Reference Architecture Model for Industry 4.0 (RAMI 4.0) standards [14], which list and catalog risk architecture levels, vulnerabilities and security issues in CPS.

A RAMI 4.0 approach helps CPSs implement a self-adaptation approach [15] that enables the system to modify its behavior and achieve its predefined performance objectives [16]. Therefore, the main goal of a self-adaptive system is to handle unexpected events such as failures, cyber threats, and undesired changes in the CPS environment. Moreover, self-adaptive systems can monitor, analyze, and plan for unexpected events, then autonomously put them in specific reaction procedures. Various approaches have been used to implement self-adaptation in control systems, such as [13] and [11]. The approach in [10] also did this, relative to only the most essential of functions, as described in the proposed MAPE-K reference model [17]. This approach investigated how anomaly detection can be used to support both monitor and analysis phases in MAPE-K.

An intrusion detection system (IDS) may include signature(-based) detection (SD), stateful protocol analysis (SPA), and anomaly(-based) detection (AD) [18]. SPA and SD can only detect known threats using signatures and rules to describe malicious events mentioned in a blacklist [19], while AD approaches can detect unknown attacks using a baseline of normal CPS behavior, known as whitelisting [20]. Indeed, rapid changes in cyber threats demand IDS self-learning approaches. Finally, AD approaches are self-learning methods that automatically and autonomously learn a system's behavior and adapt relative to these changes — providing grounds for detecting intruders.

There are three AD self-learning methods: supervised, semi-supervised, and unsupervised [21]. Unsupervised methods do not require labeled data and can distinguish between malicious and honest sources using training data. Semi-supervised methods are used when training data contain anomaly-free data only, and supervised methods are used for training sets containing normal and abnormal data.

An access control list (ACL) or ACL tokens [22]–[24] were proposed to prevent SYN attackers from placing arbitrary source addresses in their packets, sending them to different destinations. In the context of software-defined networking (SDN), [25] provided the SDN with TCP proxy anti-spoofing techniques to mitigate a spoofed SYN flood attack. However, this changes the SDN data path by adding and modifying the data-plane header. Nonetheless, [26] addressed the analysis and design of TCP handshaking, the process of forging communication with the server in the industry field, and found that neither a DoS nor a spoofed SYN flood attack were of concern. In addition, the implementation of a handshake modifies data-plane headers by adding logic, increasing delay and SDN network complexity in the industry environment. Similarly, [27] explored remotely triggered black hole (RTBH) filtering as a routing protocol for diverting DoS attack traffic, while elastic scaling using network functions virtualization (NFV) was proposed in [28]. Meanwhile, the implementation of an HTTP redirect and TCP reset as a SYN anti-spoofing approach for Industry 4.0 was described in [29].

Traceback and pushback are more sophisticated network methods for tracing spoofed source addresses to detect network attacks [30]–[33]. Traceback focuses on identifying the source of spoofed addresses; like source address filtering, it does little to avoid attacks. This is unlike DoS attacks, which use a large number of compromised machines. As such, traceback methods are invaluable in detecting and

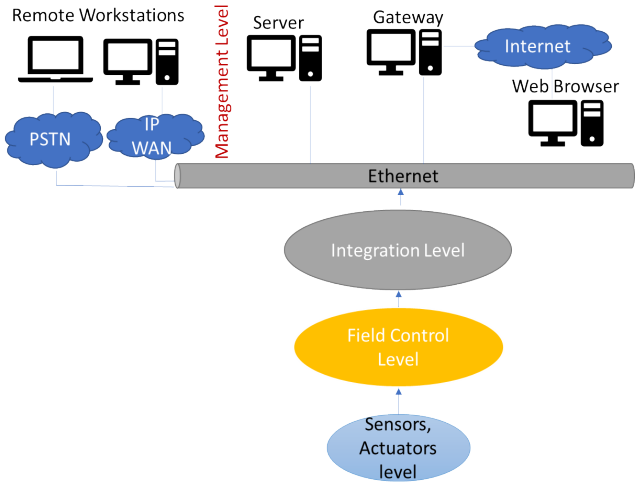

FIGURE 1. The four-level system architecture for Industry 4.0 control systems.

mitigating DoS attacks that target industry TCP communications. Conversely, pushback proposes to overcome the limitations of traceback in [34], [35], but as it is used for dynamic traffic filtering, it focuses solely on controlling link bandwidth and determines whether it is flooded.

Focusing back, AD is a statistical rule-based method of classifying traffic as either friendly or malicious [36], [37]. In the latter case, malicious attackers may cause a number of actions such as sending automatic e-mails, raising alarms, or installing network filters. To alleviate this problem, the response to spoofed addresses must be fast to avoid damage or loss of service. For PlanetLab (a test bed for networking), 10% of its sites were to immediately disconnect machines due to receiving automated e-mail messages generated through such AD systems. Therefore, using this method to mitigate spoofed addresses is insufficient. That said, if we could start over, how would we redesign the traditional MAPE-K model to resist cyber attacks within Industry 4.0?

This paper proposes a fully supervised statistical ML to detect different types of cyber threats in CPSs. All previous efforts were based on data extracted from sensors, actuators, and controllers, whereas our solution is grounded on using fully supervised statistical ML as a method of instructing the traditional MAPE-K model to gauge normal system behavior, while generating alerts upon detecting different cyber threats. These range from spoofed flooding SYN attacks targeting computing servers, to those compromising PLC ladder logic.

III. INDUSTRY 4.0 SYSTEM ARCHITECTURE FOR SELF-ADAPTIVE CPS

This section discusses Industry 4.0 system architecture to enable and help control CPSs to become self-adaptive learning systems.

A. INDUSTRY 4.0 CONTROL SYSTEM ARCHETICURE

Fig. 1 describes the proposed control system architecture for Industry 4.0 using four levels, or components. First, the

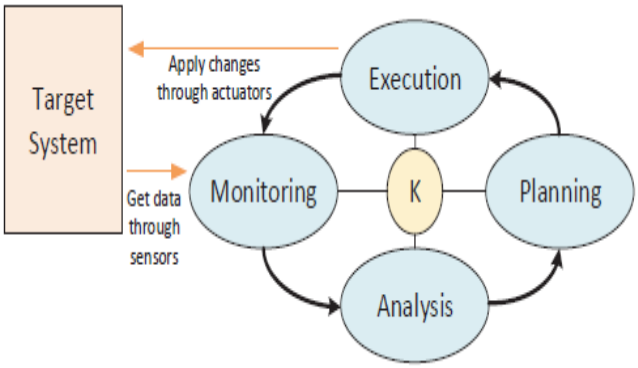

FIGURE 2. MAPE-K cycle.

management level provides the means to view and command automation system data. This level can serve as the operator workstation or computing server with appropriate software. When workstations operate in a stand-alone capacity, they can connect temporarily in a variety of ways, such as with a network address translation (NAT) firewall. When one or more operator workstations permanently connect to the system over an Ethernet connection, one acts as the system server and all others are clients. Meanwhile, integration-level components are controllers. These intelligent programmable devices work with the components at the management level to implement control strategies for an entire CPS facility. Next, the field controller level sees programmable devices support a range of applications for field control equipment, such as central plant and mechanical equipment. Finally, sensors and actuators include devices such as sensors, which can serve as both tenant control centers and field service tools. Most research studies on control systems ignore the fact that the management level is vulnerable to malicious attacks. Instead, the literature emphasizes how to mitigate such attacks and build their approaches based on data extracted from sensors and actuators levels.

B. INDUSTRY 4.0 SELF-ADAPTIVE CONTROL SYSTEM

Existing industry control systems are self-adaptive and apply feedback loops, as explained in the MAPE-K model. Essentially, this configuration integrates computational and physical components in CPSs and can be considered the crucial reference control model for automatic and self-adaptive systems [17], as shown in Fig. 2.

The 'knowledge' base represents the data related to the targeted system environment, including software control parameters, adaptation goals, sensors, and actuator data saved in the management level (as shown in Fig. 1). The monitoring phase then contains all the data collected from the sensors and actuators level, while the analysis phase determines whether adaptation is required. If so, the planning phase next carries out the necessary actions following some predefined policies, which are necessary to achieve CPS system goals. The final execution phase completes the process and performs the required actions using actuators. Together, these phases in the MAPE-K model communicate with each other in the

knowledge phase using Ethernet technology at the management level. These phases can be decentralized throughout multiple loops in the CPS.

The main function of the monitoring phase is to collect data from the physical field and forward this information onto the second phase (analysis). Herein, these data are filtered to separate the necessary information from the unnecessary information, and subsequently analyzed to determine if any adaptation is required. Therefore, if the system is operating at suboptimal conditions, modification requests are generated in the analysis phase and delivered to the planning phase.

In the planning phase one or more predefined self-adaptation policies are selected to generate a required action, which is forwarded to the execution phase. The type of predefined self-adaptation policies is based on delivered data from the analysis phase. These policies can be ECA policies, goal policies, or utility function policies [13], which are generated using automated planning.

The execution phase next executes actions delivered from the planning phase using actuators from the sensors and actuators level (see Fig. 1). Servers at the management level (as in Fig. 1) can be used to carry out actions from the planning phase and forwarded to actuators.

Evidently, the MAPE-K model is static in nature and relies on predefined policies that reside in the planning phase [10]. Therefore, if there are any new malicious attacks not previously defined in that phase, MAPE-K will fail to generate any actions to protect the CPS system. This can be considered a crucial deficiency in the current industry's control system protection. Moreover, the knowledge, planning, and execution phases reside in servers or at the management level, which can also be considered vulnerable to DoS flooding SYN attacks. According to MAPE-K model design, this type of model cannot deal with attacks that target high-level application in a server.

C. INDUSTRY 4.0 SELF-LEARNING ADAPTIVE CONTROL SYSTEM

As described in Section B, MAPE-K is a self-adaptive model that relies on predefined policies to alter CPS controls against uncertain conditions. However, the model is not dynamic and cannot, therefore, detect new cyber threats if they are not predefined in the planning phase. As it lacks CPS control system knowledge to adapt in uncertain conditions, this paper proposes a MAPE-K model that understands Industry 4.0 control systems and automatically adapts itself to detect and mitigate abnormal behavior. As shown in Fig. 3, this would mean that in the monitor phase necessary data are collected from sensors and forwarded to the learning phase. At this point, a collection of crude information follows are generated by typical VLX programs as well as numerous attacked VLX programs. The next step is to extricate positive and negative feature vectors from this information to perform supervised learning, which comprises the following steps: (1) settling a feature vector sort; (2) collecting feature vectors from the data; and (3) applying a supervised learning calculation.

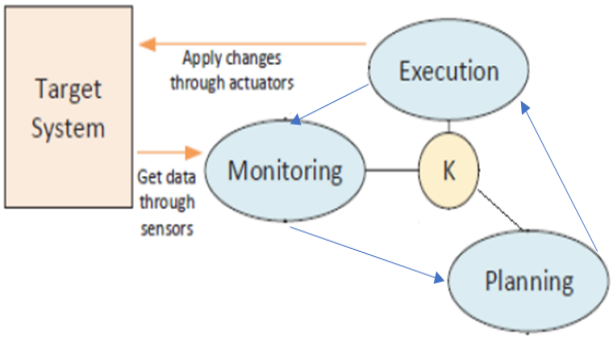

FIGURE 3. Industry 4.0 self-learning adaptive model.

First, Step (1) (feature vector sort) must be characterized to suitably speak to objects of the information. That is, for traces of sensor information, a straightforward feature vector would comprise of the sensor values at any given time point. However, for commonplace CPSs, such feature vectors are both distant and basic, since they do not typify approximately how the values evolve over time — an inborn portion of the physical model. A more valuable feature vector would record the values at fixed time interims, making it conceivable to memorize designs around how the levels of hot water, for example, alter over the time series. Within the case of a hot water system (HWS) test (described in Section IV), we defined our feature vectors to be of the shape h, h^-, where h denotes the hot water temperature at a certain time and h^- denotes the values of the same water temperature after t time units; here, t is some settled time interim of a different interval at which information is logged. The feature vectors are based on the sliding window strategy commonly utilized for time series information [38].

Step (2) (collecting feature vectors) next sees the raw normal and irregular information sorted into positive and negative feature vectors of the sort chosen in Step (1). Extricating positive feature vectors from the normal data is direct, but one can encounter trouble when dealing with their negative counterparts, in that attacked VLXs are not guaranteed to be compelling (i.e., able to create information distinguishable from typical vectors). Besides, a successfully attacked VLX may not cause a prompt alter. It is also vital not to mislabel normal information as abnormal, as additional sifting is required. Likewise, collecting positive feature vectors is exceptionally basic: all possible pairs of physical states $(h; h^-)$ are extricated from the normal traces. For each combination $(h; h^-)$ extricated from the abnormal traces, the unmodified test system is run on h for t time units. If the unmodified test system leads to a state distinguishable from h^-, the initial match is collected as a negative feature vector; in case it leads to a state that is undefined from h^-, it is disposed of (since the transformation had no effect). In the case of an HWS, its test system is deterministic, permitting for this judgment to be made effortlessly.

In Step (3) (learning), once the feature vectors are collected a statistical ML calculation can be connected to memorize

TABLE 1. HWS security metrics and learnt adaptive action schedule.

Descriptor	SM when set point changes from 180.14 °F to 140.0 °F	Learnt adaptive action when set point changes from 180.14 °F to 140.0 °F	Low alarm	Set point	High alarm
HWS (AV-100)	–		–	180.14 °F	–
OSA temp (AI-1)	SM01: No alarm (under set point)		50 °F	68 °F	95 °F
HWR temp (AI-3)	SM02: High alarm (above set point)	LAA01: Disable boiler and pump	86 °F	180.14 °F	176 °F
HWSS temp (AI-4)	SM03: High alarm (above set point)	LAA02: Close valve	86 °F	180.14 °F	176 °F
Valve status (AO-0)	SM04: NO (above open)		NC	Open	NO
Boiler status (BO-1)	SM05: Enable		Enable	Enable	Disable
Pump status (BO-0)	SM06: Enable		Enable	Enable	Disable

Hot water system supply (HWSS); hot water return (HWR); learnt adaptive action (LAA); normal close (NC); normal open (NO)

a model. For the HWS test, we applied an SVM as our supervised ML approach since it is fully automatic, with well-developed dynamic learning techniques and a good library (we utilized LIBSVM [39]). Further, SVM has expressive parts and has regularly and successfully been applied to time arrangement expectations [40]. Based on the training data, SVM endeavors to memorize the (obscure) boundary that separates it. Distinctive classification capacities exist for expressing this boundary, extending from ones that aim to discover a simple linear division between the information, to non-linear solutions based on RBFs (distinctive classification functions for HWS are discussed in Section IV). For the purpose of approving the classifier and evaluating its generalizability, it is vital to train it as if it were a parcel of the featured vectors, saving a portion of the information for testing. We arbitrarily selected 70% of the feature vectors for the preparatory set, saving the rest for evaluation, and noted that SVM can battle to memorize a reasonable classifier on the off chance that the data were exceptionally lopsided. For the case of the HWS test, one test system was delegated for ordinary data, and boundless attacked VLX test systems were for producing abnormal data. To guarantee its adjustment, we undersampled the negative feature vectors. Let M_{No} signify the number of positive feature vectors and M_{Ne} the number of negative feature vectors collected. We parceled the negative vectors into subsets of measure ($M_{Ne} = M_{No}$) (adjusted to the closest numbers) and randomly selected a feature vector from each one. As result, we can distinguish an undersampled set of negative feature vectors that was measuring as the positive set.

The next step was to approve the classifier. At this point, we had collected typical and irregular data, processed them into positive and negative vectors, and learnt a classifier by applying a directed ML approach. This comprised of the taking after two Steps: (1) applying a standard ML cross-validation to evaluate how well the classifier summed up; and (2) applying the sequential Monte Carlo (SMC) method to determine whether there was factual proof that the classifier characterized an invariant property of the framework.

Step (1) concerns cross-validation. To begin this process, a standard ML k-fold cross-validation (e.g., k = 5) was applied to survey how well the classifier summed up. This technique computes the normal exactness of k diverse classifiers, each obtained by dividing the preparatory set into k segments, training using k − 1, and approving the remaining fragment (repeating with regard to diverse approval allotments).

Step (2) next involves measurable demonstrate checking. This is when the validation method applies the SMC, a standard method for verifying general stochastic frameworks [41]. The variation utilized observed executions of the framework (i.e., follows of sensor information), and applied hypothesis testing to decide whether these implementations gave factual proof of the learnt demonstrate being an invariant of the framework. Essentially, the SMC gauges the likelihood of correctness, rather than ensures it by and large. It is basic to apply, since it must be able to execute the (unmodified) system and collect information follows. It treats the framework as a dark box and, thus, does not require a show [42].

Given a few classifiers, we applied the SMC to decide whether it was an invariant of the VLX set points with a probability more prominent or equivalent to a few edges. Hence, our model used the learning phase to generate security metrics (SM) or events after classifying data into positive and negative categories (see Table 1). It also used the negative data to generate SMs or alarms to protect and maintain CPS self-adaptation. It became clear that through a learned MAPE-K we can teach CPSs how to detect, mitigate, and learn about attacks that target servers, VLX, sensors, or actuators.

We also propose to apply statistical ML on sensor data, network IPs and MAC addresses, network TCP traffic, and a number of trials to access PLC controllers. This also concerned changes in software parameters that control physical processes to design models that characterize invariant properties (consistency). Here, conditions have to justify in all states in such processes controlled by CPSs, and carry out this consistency at runtime. Therefore, we propose statistical ML that classifies sensor data, network traffic, network IPs, and MAC addresses into both positive cases (which represent normal behavior and justify consistency) and negative cases (which represent abnormal behavior, thus, creating SMs or alarms).

IV. EVALUATION

This section evaluates a proposed solution by applying the proposal in a real CPS to control the HWS in a smart building.

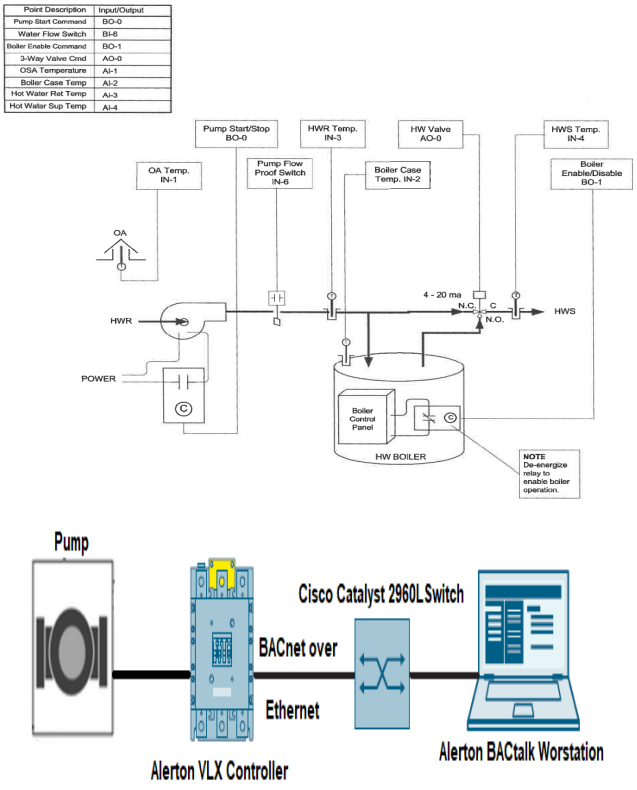

FIGURE 4. HWS architecture.

A. TESTING ENVIRONMEMNT

A real HWS in this paper was used to control water temperature in a building, notably the full operation and design of the automation system. The HWS consisted of one pump equipped with dedicated adjustable frequency controllers to adjust speed. Such systems use a chilled water-flow transmitter, bypass water-flow transmitter, building hot water temperature sensor, looped pressure differential sensor, hot water temperature control valve, building-isolation control valve, and bypass pressure control valve to adjust the pump speed for measurement and control.

As shown in Fig. 4, our system consisted of Alerton's BACtalk with Envision software for management purposes and to send control signals to the Alerton VLX controller. Both the workstation and controller interconnected through a Cisco Catalyst 2960L switch. Pump speed was controlled using a Alerton VLX controller, as configured by a laptop or by opening a hyper-terminal connection using the Alerton workstation. In addition, the VLX controller interconnected with the workstation through a Cisco switch using a BACnet over IP protocol.

B. HWS PROCESS IN A SMART BUIDLING

1) HEATING SYSTEM CONTROL

The process began by starting the hot water circulation pump. After five minutes of heating the pump, the boiler's self-contained temperature controls were enabled. Following, the pump was stopped and the boiler disabled when there was no longer a demand for heating. The pump continues to operate for five minutes to dissipate residual heat in the boiler.

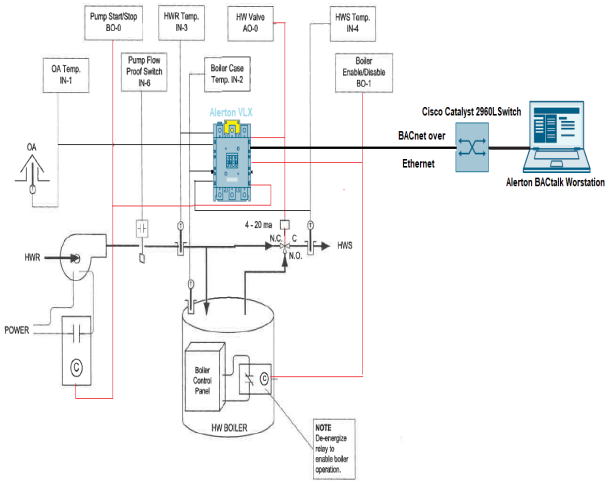

FIGURE 5. VLX input and output.

In our HWS, we represented all input and output logic points on an operator workstation and VLX controller [43], as follows:

- Inputs and outputs (analog input [AI], analog output [AO], binary input [BI], and binary output [BO]): AI and BI are associated with electrical physical inputs connected to VLX, but AO and BO are associated with electrical physical outputs connected to the VLX controller.
- Values (analog value [AV], binary value [BV]): these values are used as calculated values in VLX or the operator workstation. Set points, timers, or any virtual value are not associated with electrical physical inputs or outputs.
- AV-100 can be interpreted as AI-0 and BO-0 (etc.) uniformly.

2) SECURITY METRICS OR SYSTEM ALARMS

Alarm indications at the operator's workstation are specified for:

1) heating water fails (latch alarm and provide reset alarm)
2) heating water pump maintenance (set point adjustable; 500–1500 H)
3) boiler maintenance (set point adjustable; 500–1500 H).

Other indication alarms are included at the operator's workstation including indication of HWSS temperature, HWR temperature, boiler case temperature, outside air temperature, and boiler-enable status (among others).

Typically, Alerton workstations use Microsoft Visio software to configure and manage the VLX controller. This is programed to enable or disable software logic points such as BO to start or stop the pump, relying on specific configuration conditions.

Fig. 5 shows the input of the VLX controller to control HWS pump speed and maintain the aforementioned

TABLE 2. SVM-based classifier functions comparison.

Type	Accuracy	Cross-validation accuracy	Sensitivity	Specificity
SVM-linear	73.34%	74.12%	73.44%	69.23%
SVM-polynomial	78.10%	79.32%	78.92%	71.67%
SVM-RBF	95.15%	94.99%	99.55%	92.82%

HWS processes. Next, the VLX logic is configured and uploaded using software tools on the Alerton workstation. Its output can assume both BOs and AOs to enable or disable the boiler and open or close the valve, respectively (see Fig. 5). This process reliies on (i) the water level inside the boiler, (ii) demand from users inside the building, and (iii) the temperature sensors connected to the VLX. HWR and HWSS sensors, the boiler case, and OSA send their values every second to VLX, relying on input values deriving from temperature sensors with the software logic point (AI) on VLX. As such, the HWS's three-way valve will NO, NC, or close when the system's set point changes (AV-100).

The HWS set point will reach 180.13 °F (82.3 °C) if the temperature on the OSA sensor is 68 °F (20 °C). If the OSA is lower than 68 °F, VLX enables the pump logical point (BO-0); here, the three-way value will be NO (AO-1) and the boiler will be enabled (BO-1). This process continues until the temperature inside the boiler reaches AV-100 (equal to 180.13 °F). If the OSA is higher than 68 °F, VLX disables the pump logical point (BO-0); now, the three-way value will be NC (AO-1) and the boiler will be disabled (BO-1).

Trend logs are used to achieve the first stage in the learned MAPE-K model (i.e., the monitor phase). By using the statistical ML (SVM), our automation process learns if the logic of the VLX and workstation are alerted and if there are invariant or consistent conditions; likewise, potential corruption of either element is also indicated. This can come in the form of a SYN flooding attack that affects network traffic and changes the control values on the workstation and VLX. As such, two types of corruption were examined. The first was a network attack (e.g., SYN flooding attack), which is created using the hping3 tool [44] to generate SYN packets with different variants, such as source IP, source port, and packets per second (PPS) rates. The second corruption type concerned changes in VLX control values, which lead to physical condition variants.

C. NETWORK ATTACK: OPERATOR WORKSTATION UNDER DoS ATTACKS

In CPSs, intruders aim to target the Cisco switch using a spoofed SYN attack to connect to the Alerton server, prompting a server crash. Therefore, it was important to learn spoofed SYN flooding in our HWS automation process to evade this vulnerability. The operator workstation sends the runtime logs charting network traffic, as well as the VLX input and output values to another workstation, whose parameters necessitate a MacBook (Intel) running at 2.20 GHz, with 2.00 GB of RAM. In spoofed SYN flooding attacks, hping3 is used to send multiple SYN requests to the server or operator workstation using a remote station to create a TCP connection, but with different random source MAC and IP addresses. The main purpose of an intruder is to flood a server with SYN requests to prompt it or the operator workstation to crash. Training data pairs (as in Section III-C) are next created by SVMs as supervised ML. However, as our first experiment was to determine which of the SVM-based classification functions — linear, polynomial or RBF — should be used to instil our model with a high level of accuracy, it proved necessary to first generate 900 TCP requests using hping3 with 101 (or abnormal behavior). For this, we used undersampling to generate feature vectors, which randomly divided into 70% for training and 30% for testing feature vectors. SVM as a supervised ML was also implemented by MATLAB 9.1.0 and subsequently applied to the training vectors to learn three separate linear, polynomial, and RBF classifier functions.

The first experiment is described in Table 2, which presents a comparison between linear, polynomial, and RBF classifiers learnt using SVM. 'Accuracy' reports how many testing feature vectors are labeled correctly, and 'cross-validation accuracy' is the average accuracy of five different classifiers by dividing the training set to five, training four partitions and validating on the fifth. The benefit of cross-validation accuracy is that it measures the degree of generalization in our classifier. Next, 'sensitivity' measures the proportion of positive values, which are correctly labeled, while 'specificity' measures the negatives. Across all four measures, the classifier (which has a higher percentage) was superior and, thus, was selected.

From Table 2, it is clear that the RBF classifier (99% average) outperformed the other two classifiers (linear and polynomial, 69–79%). Hence, it is believed that linear and polynomial classifiers are not sufficient because the TCP SYN segments, which are beyond the expressiveness of both types of classifier, correlate.

To assess the effect on accuracy using different round-trip times (RTT) for each SYN segment feature vector, we determined the time interval by Υ (vector interval between the remote workstation. This creates the hping3 and operator workstation, or the Alerton server.

In SYN spoofed attacks, source IP addresses are different and CPU usage is very high. Due to saturation of the Cisco switch control channel during such instances, the Ethernet

TABLE 3. Effect of RTT time interval on stability of SVM-RBF classifier.

Time interval	Accuracy	Cross-validation accuracy
20	93.34%	90.22%
25	93.10%	93.08%
30	93.20%	93.12%
35	94.95%	93.98%
40	93.70%	93.98%

network becomes vulnerable to DoS attacks, which affect automation system operations by encouraging the server to generate more SYN segments in return. As such, a second experiment warranted application of SVM classifiers on different time intervals using RTT values between the remote station and the Alerton operator workstation. The remote station then sent TCP SYN requests with high data rates from 100 kpps to 200 kpps to the Alerton server using hping3. We measured RTT for segments generated by hping3 and timed the interval at 40 μs using Wireshark software (version 2.6.1). Any effects on accuracy were next assessed using different RTT values in the feature vectors (20 μs and 40 μs). With these, we characterized the effect of spoofed SYN flooding attacks during both intervals and discovered that attacker behavior was more observable for larger RTTs, with more spoofed SYN segments received. However, if the interval was too big, this increased the number of spoofed SYN segments and strengthened the attackers' position by sending more spoofed segments, thus, crashing the server. It became apparent that using an SVM-RBF classifier to detect attackers in very short time intervals was crucial. Table 3 represents the comparison between accuracy and across-validation accuracy using different times, starting from 20 μs, 25 μs, 30 μs, 35 μs, and 40 μs. SVM-RBF was used as a classification function and hping3 was used to generate 900 spoofed SYN segments with spoofed IP addresses at different time intervals between 20 μs and 40 μs. Evident in Table 2, the SVM-RBF classification function was a more stable measure to detect attackers, especially at 35 μs. Therefore, our model shows high stability and accuracy at 35 μs, which was selected accordingly.

D. ALERTON VLX CONTROLLER UNDER ATTACK

This section characterizes the behavior of any intruders targeting the VLX ladder logic in our HWS automation system. Their main goal was to cause damage in the system by compromising the values of the AV-100, affecting BO-0, BO-1, and AO-0 (pump and boiler enable/disable, and three-way valve control software points [NO, NC, or 'close'], respectively) (see Fig. 5). The intruders in the HWS aimed to increase the speed of the pump to elevate the water level inside the boiler; this raised the pressure inside the boiler, causing boiler exposure. In addition, the intruders wished to enable the boiler and run it for an extended time to increase water temperature above 180.14 °F — the HWS's (AV-100) set point.

TABLE 4. Results of detecting VLX attacks.

Object	Description	Default	Attack	Detect	Accuracy
AV-100	HWS temp set point	180.14 °F	140 °F	Yes	91.34%
BO-0	Pump status	Off	On	Yes	98.05%
BO-1	Boiler status	Off	On	Yes	97.56%
AO-0	Valve signal	NC	Close	Yes	98.57%

Moreover, the intruders aimed to change the status of the three-way valve control from NO to 'close' to increase pump pressure, leading to its exposure. Therefore, the intruders intended to attack the Alerton vision software operator workstation to change the VLX control sequences. Attackers typically use many tools to access and change logic such as mutant codes [45] or DarkComet [46]. Upon accessing the HWS automation network, they can subsequently modify the VLX logic and upload the alerted configuration settings.

VLX monitors temperature input points (AI-3, AI-4) from temperature transmitter sensors furnished with a domestic HWS to create alarms and enable or disable the boiler and pump, or even open or close the three-way valve. If intruders manage to change the HWS (AV-100 = 180.14 °F) set point, this will lead to a reverse operation of the pump, boiler, and three-way valve, as shown in Table 1. For example, if intruders changed the HWS set point to 140.54 °F this would speed up the pump and enable the boiler for an extended period, increasing the water temperature above its normal point. Also, the alarm will take some time to sound, thus, affecting HWS operation. As a result, water temperature inside the boiler will overheat and the HWS will not (and cannot) detect VLX logic changes. Consequently, the SVM-RBF classifier studies the CPS using HWR and HWSS sensor readings (which obtain the real and actual values without any compromised actions) to create security metrics SM01, 02, 03, 04, 05, and 06, mapping to actions LAA01 and 02 (see Table 1).

As such, the final experiment assessed whether the learned MAPE-K model could detect different types of VLX attacks by classifying the feature vectors as negative, in case of attack. In this experiment, we investigated VLX code modification attacks by randomly changing VLX logic and monitoring physical effects in the HWS. There was no benchmark code to apply for logic modification, so logic in the VLX controller was designated using Alerton vision software. If the SVM-RBF can generate an alarm relying solely on latter attack detection, the consistency of the HWS can be regarded physical proof of the integrity of VLX logic.

Table 4 depicts a list of VLX logic modification attacks applied for the HWS in the operator workstation; this includes the results of our consistency attempts at classifying them. The attacks intended to change logic values for the pump, boiler, valve, and the software point status (AV-100) to affect HWS operation. According to Table 4, the SVM-RBF classifier can detect attacks and accurately label the feature vectors as negative, thus, indicating an attack. If the accuracy is

higher than 90% (which is very high), an alarm will sound and demonstrate accurate detection. Evidently, 18 of 20 mutant codes affected AV-100, six of eight affected BO-0 and BO-1, and four of four affected AO-1. It is believed that undetectable mutants are due to undersampling, which converts logic values from abnormal to normal.

V. CONCLUSION

CPS is the technical driving force behind transformation in the production toward digital applications and Industry 4.0. However, such a system creates a crucial concern regarding security, detection, and mitigation, as attackers and software become more intelligent, interconnected, and intertwined over time. This paper illustrated how CPSs can benefit from the proposed MAPE-K model to become more self-learned. Overall, its evaluation on a real HWS system indicated its capability to detect and mitigate network and VLX attacks with high accuracy.

REFERENCES

[1] N. Jazdi, "Cyber physical systems in the context of Industry 4.0," in *Proc. IEEE Int. Conf. Automat., Qual. Test., Robot.*, May 2014, pp. 1–4.
[2] i-SCOOP. *Industry 4.0: The Fourth Industrial Revolution Guide to Industry 4.0*. Accessed: Apr. 10, 2018. [Online]. Available: https://www.i-scoop.eu/industry-4-0/
[3] Industrial Internet Consortium, *Industrial Internet of Things. Volume G4: Security Framework*, 2016.
[4] K. Stouffer, J. Falco, and K. Scarfone, "Guide to industrial control systems (ICS) security," *NIST Special Publication*, vol. 800, no. 82, p. 16, Jun. 2011.
[5] S. Han, M. Xie, H.-H. Chen, and Y. Ling, "Intrusion detection in cyber-physical systems: Techniques and challenges," *IEEE Syst. J.*, vol. 8, no. 4, pp. 1052–1062, Dec. 2014.
[6] W. Eddy, *TCP SYN Flooding Attacks and Common Mitigations*, document RFC 4987, Aug. 2007.
[7] *Stateful Connection Tracking Stateful NAT*. Accessed: Jan. 20, 2019. [Online]. Available: http://openvswitch. org/support/ovscon2014/17/1030 conntrack nat.pdf/
[8] G. Bianchi, M. Bonola, A. Capone, and C. Cascone, "OpenState: Programming platform-independent stateful openflow applications inside the switch," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 2, pp. 44–51, Apr. 2014.
[9] S. W. Shin and G. Gu, "Attacking software-defined networks: A first study," in *Proc. HotSDN*, Aug. 2013, pp. 165–166.
[10] G. Settanni, F. Skopik, A. Karaj, M. Wurzenberger, and R. Fiedler, "Protecting cyber physical production systems using anomaly detection to enable self-adaptation," in *Proc. IEEE Ind. Cyber-Phys. Syst. (ICPS)*, May 2018, pp. 173–180.
[11] M. Tauber, G. Kirby, and A. Dearle, "Self-adaptation applied to peer-set maintenance in chord via a generic autonomic management framework," in *Proc. 4th IEEE Int. Conf. Self-Adapt. Self-Organizing Syst.*, Sep. 2010, pp. 9–16.
[12] M. Hankel and B. Rexroth, "The reference architectural model industrie 4.0 (RAMI 4.0)," in *Proc. ZVEI*, Apr. 2015, pp. 1–2.
[13] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, Jan. 2003.
[14] Z. Ma, A. Hudic, A. Shaaban, and S. Plosz, "Security viewpoint in a reference architecture model for cyber-physical production systems," in *Proc. IEEE Eur. Symp. Secur. Privacy Workshops*, Apr. 2017, pp. 153–159.
[15] H. Muccini, M. Sharaf, and D. Weyns, "Self-adaptation for cyber-physical systems: A systematic literature review," in *Proc. 11th Int. Symp. Softw. Eng. Adapt. Self-Manag. Syst.*, May 2016, pp. 75–81.
[16] A. Musil, J. Musil, D. Weyns, T. Bures, H. Muccini, and M. Sharaf, "Patterns for self-adaptation in cyber-physical systems," in *Proc. Multi-Disciplinary Eng. Cyber-Phys. Prod. Syst.*, May 2017, pp. 331–368.
[17] P. Arcaini, E. Riccobene, and P. Scandurra, "Modeling and analyzing MAPE-K feedback loops for self-adaptation," in *Proc. 10th Int. Symp. Softw. Eng. Adapt. Self-Manag. Syst.*, May 2015, pp. 13–23.
[18] H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, and K.-Y. Tung, "Intrusion detection system: A comprehensive review," *J. Netw. Comput. Appl.*, vol. 36, no. 1, pp. 16–24, 2013.
[19] M. E. Whitman and H. J. Mattord, *Principles of Information Security*. 4th ed., Stamford, SA, USA: 2012.
[20] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Macia-Fernandez, and E. Vazquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," *Comput. Secur.*, vol. 28, nos. 1–2, pp. 18–28, 2009.
[21] M. Goldstein and S. Uchida, "A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data," *PloS One*, vol. 11, no. 4, 2016, Art. no. e0152173.
[22] P. Ferguson and D. Senie, *Network Ingress Filtering: Defeating Denial of Service Attacks that Employ IP Source Address Spoofing*, document RFC 2827, 2000.
[23] N. Lu, S. Su, M. Jing, and J. Han, "A router based packet filtering scheme for defending against DoS attacks," *China Commun.*, vol. 11, no. 10, pp. 136–146, Oct. 2014.
[24] F. Soldo, A. Markopoulou, and K. Argyraki, "Optimal filtering of source address prefixes: Models and algorithms," in *Proc. IEEE INFOCOM*, Apr. 2009, pp. 2446–2454.
[25] S. Shin, V. Yegneswaran, P. Porras, and G. Gu, "AVANT-GUARD: Scalable and vigilant switch flow management in software-defined networks," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2013, pp. 413–424.
[26] A. Ferguson, A. Guha, C. Liang, R. Fonseca, and S. Krishnamurthi, "Participatory networking: An API for application control of SDNs," in *Proc. ACM SIGCOMM Conf.*, Aug. 2013, pp. 327–338.
[27] K. Giotis, G. Androulidakis, and V. Maglaris, "Leveraging SDN for efficient anomaly detection and mitigation on legacy networks," in *Proc. 3rd Eur. Workshop Softw. Defined Netw.*, Sep. 2014, pp. 85–90.
[28] S. Fayaz, Y. Tobioka, V. Sekar, and M. Bailey, "Bohatei: Flexible and elastic DDoS defense," in *Proc. 24th USENIX Secur. Symp.*, Aug. 2015, pp. 817–832.
[29] Y. Afek, A. Bremler-Barr, and L. Shafir, "Network anti-spoofing with SDN data plane," in *Proc. IEEE INFOCOM*, May 2017, pp. 1–9.
[30] S. Bellovin. (2000). *ICMP Traceback Messages*. [Online]. Available: http://www.research. att.com/ smb/papers/draft-bellovin-itrace-00.txt
[31] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Practical network support for IP traceback," in *Proc. Conf. Appl., Technol., Architectures, Protocols Comput. Commun.*, Oct. 2000, pp. 295–300.
[32] A. Snoeren, C. Partridge, L. A. Sanchez, C. E. Jones, F. Tchakountio, S. T. Kent, and W. T. Strayer, "Hash-based IP traceback," in *Proc. ACM SIGCOMM Comput. Commun. Rev.*, Aug. 2001, pp. 3–14.
[33] I. Priescu, S. Nicolaescu, and I. Bica, "Design of traceback methods for tracking DoS attacks," in *Proc. Int. Assoc. Comput. Sci. Inf. Technol.-Spring Conf.*, Apr. 2009, pp. 117–121.
[34] R. Mahajan, S. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker, "Controlling high bandwidth aggregates in the network," *Comput. Commun. Rev.*, vol. 32, no. 3, pp. 62–73, Jul. 2002.
[35] J. Ioannidis and S. Bellovin, "Implementing pushback: Router-based defense against DDoS attacks," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, Oct. 2002, pp. 1–8.
[36] Mazu Networks. (2003). *Self-Optimizing Network Traffic Security*. [Online]. Available: http: //www.mazunetworks.com/nts.html
[37] D. Golait and N. Hubballi, "Detecting anomalous behavior in VoIP systems: A discrete event system modeling," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 3, pp. 730–745, Mar. 2017.
[38] T. G. Dietterich, "Machine learning for sequential data: A review," in *Proc. IAPR Int. Workshops Stat. Techn. Pattern Recognit. (SPR) Struct. Syntactic*, Aug. 2002, pp. 15–30.
[39] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, p. 27, 2011.
[40] N. I. Sapankevych and R. Sankar, "Time series prediction using support vector machines: A survey," *IEEE Comput. Intell. Mag.*, vol. 4, no. 2, pp. 24–38, May 2009.
[41] E. M. Clarke and P. Zuliani, "Statistical model checking for cyber-physical systems," in *Proc. Int. Symp. Automated Technol. Verification Anal.*, Oct. 2011, pp. 1–12.
[42] K. Sen, M. Viswanathan, and G. Agha, "Statistical model checking of black-box probabilistic systems," in *Proc. Int. Conf. Comput. Aided Verification*, Jul. 2004, pp. 202–215.
[43] Accessed: Mar. 20, 2019. [Online]. Available: www.alerton.com
[44] Accessed: Apr. 20, 2019. [Online]. Available: http://linux.die.net/man/8/hping3/

[45] Y. Chen, C. M. Poskitt, and J. Sun, "Learning from mutants: Using code mutation to learn and monitor invariants of a cyber-physical system," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2018, pp. 648–660.
[46] M. Ussath, D. Jaeger, F. Cheng, and C. Meinel, "Advanced persistent threats: Behind the scenes," in *Proc. Annu. Conf. Inf. Sci. Syst. (CISS)*, Mar. 2016, pp. 181–186.

IBRAHIM ELGENDI (M'16) received the Ph.D. degrees in information technology from the University of Canberra, Australia, in 2016 and 2018. He is currently a Lecturer in network engineering. He is a member of the Institution of Engineers Australia. He has over 14 refereed publications with over 40 citations (H-index 4) in highly prestigious journals, and conference proceedings. His research interests include mobile and wireless networks, Industry Internet-of-Things (IIoT), machine learning, cyber-physical-security, data analytics, and automation. He has secured over 80 million dollars in competitive research funding by the Commonwealth and State Governments, Department of Defence, and the industry. He has served as a Reviewer for a number of journals, such as the *IEEE Wireless Communications Magazine*, the IEEE TRANSACTION ON MOBILE COMPUTING, the IEEE/ACM TRANSACTIONS ON NETWORKING, and the IEEE SYSTEM JOURNAL. He has 17 years of experience from industry in mechatronics, such as IIoT, CPS, and automation.

MD. FARHAD HOSSAIN received the B.Sc. and M.Sc. degrees in electrical and electronic engineering (EEE) from the Bangladesh University of Engineering and Technology (BUET), Dhaka, Bangladesh, in 2003 and 2005, respectively, and the Ph.D. degree from the School of Electrical and Information Engineering, The University of Sydney, Australia, in 2014. He is currently a Professor with the Department of EEE. He is also an Electrical and Electronic Engineering Consultant. He has published over 65 refereed articles in highly prestigious journals and conference proceedings. His research interests include cellular networks, the IoT, M2M communications, underwater communications, smart grid communications, and wireless sensor networks. He was a recipient of the Best Paper Award in three international conferences and the Student Travel Grant in the IEEE Global Communications Conference (GLOBECOM), Anaheim, CA, USA, 2012. He also received two awards, namely TransGrid Prize and SingTel Optus Prize in the category of Best Research Project in the Research Conversazione held in The University of Sydney, Australia, in 2012 and 2010, respectively. He has been serving as a TPC member and a reviewer in many reputed international journals and conferences.

ABBAS JAMALIPOUR (S'86–M'91–SM'00–F'07) received the Ph.D. degree in electrical engineering from Nagoya University, Japan. He is currently a Professor of ubiquitous mobile networking with The University of Sydney, Australia. He is a Fellow of the Institute of Electrical, Information, and Communication Engineers (IEICE) and the Institution of Engineers Australia. He has authored nine technical books, 11 book chapters, over 450 technical papers, and five patents, all in the area of wireless communications. He is an ACM professional member and an IEEE Distinguished Lecturer. He was a recipient of a number of prestigious awards, such as the 2016 IEEE ComSoc Distinguished Technical Achievement Award in Communications Switching and Routing, the 2010 IEEE ComSoc Harold Sobol Award, the 2006 IEEE ComSoc Best Tutorial Paper Award, and the 15 Best Paper Awards. He is an elected member of the Board of Governors, the Executive Vice-President, the Chair of the Fellow Evaluation Committee, and the Editor-in-Chief of the MOBILE WORLD, the IEEE Vehicular Technology Society. He was the Editor-in-Chief of the IEEE WIRELESS COMMUNICATIONS, the Vice President of conferences, and a member of the Board of Governors of the IEEE Communications Society, and has been an Editor for several journals. He has been the General Chair or Technical Program Chair for a number of conferences, including the IEEE ICC, GLOBECOM, WCNC, and PIMRC.

KUMUDU S. MUNASINGHE received the Ph.D. degree in telecommunications engineering from The University of Sydney. He is currently an Associate Professor in network engineering and the Leader of the IoT Research Group, Human Centred Research Centre, University of Canberra. His research interests include next-generation mobile and wireless networks, the Internet of Things, green communication, smart grid communications, and cyber-physical security. He has over 100 refereed publications with over 800 citations (H-index 16) in highly prestigious journals, conference proceedings, and two books to his credit. He has secured over U.S. $1.6 million dollars in competitive research funding by winning grants from the Australian Research Council (ARC), the Commonwealth and State Governments, Department of Defence, and the industry. He has also won the highly prestigious ARC Australian Postdoctoral Fellowship, served as the Co-Chair for many international conferences, served as an Editorial Board member for a number of journals. His research has been highly commended through many research awards including two VC's Research Awards and the three IEEE Best Paper Awards.