# Outlier Dirichlet Mixture Mechanism: Adversarial Statistical Learning for Anomaly Detection in the Fog

Nour Moustafa, *Member, IEEE*, Kim-Kwang Raymond Choo, *Senior Member, IEEE*, Ibrahim Radwan, *Member, IEEE*, and Seyit Camtepe, *Senior Member, IEEE*

***Abstract*— Current anomaly detection systems (ADSs) apply statistical and machine learning algorithms to discover zero-day attacks, but such algorithms are vulnerable to advanced persistent threat actors. In this paper, we propose an adversarial statistical learning mechanism for anomaly detection, outlier Dirichlet mixture-based ADS (ODM-ADS), which has three new capabilities. First, it can self-adapt against data poisoning attacks that inject malicious instances in the training phase for disrupting the learning process. Second, it establishes a statistical legitimate profile and considers variations from the baseline of the profile as anomalies using a proposed outlier function. Third, to deal with dynamic and large-scale networks such as Internet of Things and cloud and fog computing, we suggest a framework for deploying the mechanism as Software as a Service in the fog nodes. The fog enables the proposed mechanism to concurrently process streaming data at the edge of the network. The ODM-ADS mechanism is evaluated using both NSL-KDD and UNSW-NB15 datasets, whose findings indicate that ODM-ADS outperforms seven other peer algorithms in terms of accuracy, detection rates, false positive rates, and computational time.**

***Index Terms*— Adversarial statistical/machine learning, outlier detection, Dirichlet mixture model, anomaly detection, fog computing.**

## I. Introduction

INTERNET of Things (IoT) is increasingly popular in both developed and developing nations, and has applications even in military and battlefield environments [1]. To ensure the security of IoT and the underpinning systems, statistical and machine learning algorithms have been used in anomaly detection systems (ADSs) and other security systems (e.g., intrusion prevention/detection systems and data loss prevention systems). There has also the move towards the cloud, in order to meet the computational requirements (e.g., storage and processing capabilities) [2]. However, a cloud-based deployment has a number of associated limitations, such as high latency, low mobility support, general lack of poor location awareness and geo-distribution [3]–[5]. One potential solution is to move some computation resources closer to the users, such as in a fog computing architecture. In the fog, data is received and processed at the edge devices.

Depending on the requirements of the applications, the edge-processed data are either returned to the IoT devices or aggregated and sent to the cloud for further processing and/or long-term storage [2], [3], [5]–[8]. While cloud service providers undertake different security measures such as authentication systems, access control and intrusion prevention/detection systems [5], [9]–[11], it is unrealistic to secure all potential attack vectors. For example, vulnerabilities in any of the underlying technologies, such as network systems, data centers and virtual machines (VMs), can be exploited to target a cloud architecture [12]. The latter comprises three layers, namely: interdependent infrastructure, application and platform, and any of these layers can be targeted. In addition, protecting against zero-day adversaries of IoT networks and statistical/machine learning based ADS techniques are ongoing research and operational challenges.

Cyber attackers use different techniques to identify and exploit systems and other vulnerabilities, in order to evade detection and circumvent existing countermeasures. There have also been attempts to influence (or mislead) the performance of machine learning-based solutions by injecting malicious data used in their training [2], [3], [13]. This is also known as adversarial machine learning, where the adversary seeks to compromise the learning process of machine learning-based solutions [13], [14]. Evasion and poisoning attacks are two popular approaches to compromise learning models [13]. In the evasion attack, for example, the adversary generates one of more bugs (or flaws) in the learning process to facilitate future attacks by creating misclassification when running the model. In the poisoning attack, the adversary attempts to

Manuscript received September 25, 2018; revised December 15, 2018; accepted December 28, 2018. Date of publication January 3, 2019; date of current version May 9, 2019. The work of K.-K. R. Choo was supported by the Cloud Technology Endowed Professorship. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Xiaodong Lin. *(Corresponding author: Kim-Kwang Raymond Choo.)*

N. Moustafa is with the School of Engineering and Information Technology, University of New South Wales at ADFA, Canberra, ACT 2612, Australia (e-mail: nour.moustafa@unsw.edu.au).

K.-K. R. Choo is with the Department of Information Systems and Cyber Security, The University of Texas at San Antonio, San Antonio, TX 78249-0631 USA, also the Department of Electrical and Computer Engineering, The University of Texas at San Antonio, San Antonio, TX 78249-0631 USA, and also with the School of Information Technology & Mathematical Sciences, University of South Australia, Adelaide, SA 5095, Australia (e-mail: raymond.choo@fulbrightmail.org).

I. Radwan is with the College of Engineering and Computer Science, The Australian National University, Canberra, ACT 0200, Australia, and also with the College of Business and Economics, The Australian National University, Canberra, ACT 0200, Australia (e-mail: ibrahim.radwan@anu.edu.au).

S. Camtepe is with CSIRO Data61, Marsfield, NSW 2122, Australia (e-mail: seyit.camtepe@data61.csiro.au).

Digital Object Identifier 10.1109/TIFS.2018.2890808

modify legitimate data in the training phase to degrade the model's performance.

In this paper, we present an autonomous anomaly detection-based adversarial statistical learning mechanism designed to detect zero-days attacks from dynamic networks, even when poisoning attacks occur during the training phase. The proposed mechanism addresses the following three challenges.

- Constructing a comprehensive profile of normal activities is extremely challenging due to factors such as the dynamic nature of the network and evolution of attacker behaviors [9], [15], and injection of malicious instances such as in poisoning attacks.
- Designing an adaptable decision engine (DE) to efficiently distinguish between normal and malicious behaviors in large, high-speed network environments, is also challenging [3], [4], [6], [16]. Such networks generally comprise thousands of interconnected devices and machines (e.g., VMs and platforms), located in multiple locations (e.g. different nations).
- Ensuring the security solution is capable of handling large-volume, high-velocity and high-dimensionality of data (i.e., big data) [5], [17], [18].

To address the challenges above, we propose an Outlier Dirichlet Mixture (ODM-ADS) mechanism to efficiently discover malicious activities from dynamic and large-scale networks. We use learning statistical theories to estimate the density of the distribution of a Dirichlet Mixture Model (DMM) [19], [20] for obtaining a profile of the legitimate training set. For adversarial learning, the training set is injected by data poisoning attacks in which some malicious instances are trained as legitimate ones. A decision-making method is developed for constructing a baseline profile of normal data by considering the injected malicious instances. We determine to what extent the injection of malicious instances could degrade the performance of the model that can deal with high dimensional data. Moreover, the model contains a generalized distribution that deals with outliers like malicious instances inserted in the training phase. We also propose a framework for deploying the proposed mechanism as Software as a Service (SaaS) to address the drawbacks of running intrusion detection at the cloud. The performance of the ODM-ADS mechanism is assessed on two popular datasets, the NSL-KDD [21], [22] that is an enhanced version of the KDD99 dataset widely used to evaluate new NIDSs, and the UNSW-NB15 [23]–[25] which consists of many recent normal and attacks vectors.

The key contributions of this paper are in the following:

- We present statistical measures that can be used to define the potential properties of network data, i.e., data normality and linearity, in order to determine the best DE approach, i.e., an autonomous and lightweight ADS.
- We propose a new DE module, hereafter referred to as the ODM-ADS mechanism, for detecting suspicious activities along with handling adversarial behaviors in dynamic networks.
- To mitigate limitations in a cloud-based deployment, we present a fog-based ODM-ADS framework.

The remainder of this paper is structured as follows. We briefly introduce relevant background and related studies in Section II. The proposed ODM-ADS mechanism is presented in Section III, and the fog-based framework is introduced in Section IV. We then present the evaluation in Section V, prior to concluding the paper in the last section.

## II. Background and Related Work

We review related background and prior work, particularly those focusing on intrusion detection systems (IDS) and adversarial machine learning for cloud and fog computing systems.

### A. Adversarial Machine Learning and IDS

Attacks can take place at various stages in a machine learning-based security solution, such as in the data inputs, model structure, training and testing phases, parameters, feature extraction, and/or model outputs [26]–[28]. For example, to influence the design of machine learning algorithms and their input [13], the attackers can seek to compromise the model structure, such as during the training and testing phases, parameters' tuning and expose datasets. Integrity-related attacks include attempts to alter and/or inject malicious activity to the training and/or testing datasets. Availability-related attacks include attempts to prevent legitimate users from accessing the models' processing while extracting features and/or inferring actions of suspicious events [14], [29].

Existing statistical and machine techniques still suffer from the lack of a generalized architecture that could assist in understanding the semantics of learning processes. This is because the security goals of the CIA (confidentiality, integrity and availability) principles have not been considered while training and validating learning models [14], [29]. In other words, in the field of adversarial machine learning, we need to study the trustworthiness of the machine learning algorithms that are subject to different sophisticated attacks [13], [14]. For example, how do we adequately deal with adversaries attempting to discover weak stochastic properties and dynamic data distributions of learning models?. Can we use adversarial machine models to facilitate the understanding of how attack strategies could be identified and mitigated?.

In order to evaluate the robustness of learning models, the adversarial learning of poisoning attacks could be evaluated using an IDS, which monitors and inspects activities taking place in a host or network system to discover possible threats [3], [12], [15], [16]. IDS methods can be broadly classified into three types, namely: misuse-, anomaly- and hybrid-based, with the last a combination of the first two [2], [3], [9], [15]. A misuse-based detection system (MDS) monitors network traffic or hosts / virtual machines (VMs) to match observed activities with attack signatures logged in a database. However, the MDS is ineffective against zero-day attacks. Also, significant effort is required by security experts to regularly update its database in a cloud computing environment, which includes a (large) number of rules regarding malicious events [3], [15], [17].

An ADS establishes a normal profile and identifies any variation from it as an attack. It can identify both known

and zero-day malicious activities, and requires less effort than a MDS to create its profile [3], [15], [30], [31]. A typical ADS consists of four key modules, a data source, data preprocessing, DE and security response [15], [23], [32]. A data source includes host and/or network data for enabling the DE to discover abnormal instances [15], [18], [28]. The data preprocessing module is a significant part of learning theories as it processes and filters the input data by excluding irrelevant attributes. The DE module is obviously critical in the design of an efficient model for identifying malicious instances [32]. Ultimately, the security response module indicates a decision taken to stop an intrusive event [15].

An ADS in the cloud can discover malicious events in either real- or non real-time [3], [33]. In the former, attacks are detected when the hosts, VMs or network are being examined, with any variations from normal activities immediately flagged as anomalies. In the latter, a non real-time detection module handles traced data, which can be aggregated in a centralized manner from either a single or distributed system(s) of multiple network nodes, with a delay. A VM monitor inserted as a software layer to manage the physical resources is one of several solutions, which can be implemented in many operating systems. It can improve the efficiency of detecting and preventing attacks in IDS because it can control a system's resources, including continuously monitoring its VMs' internal states [16].

Fog computing, coined in 2015 by the OpenFog Consortium, is an architecture that extends the main functions of cloud computing to provide services at the edge of a network. It helps to address key drawbacks of cloud computing systems, such as lack of low latency, mobility support, location awareness and geo-distribution [4]. An IDS can be placed at the fog side to recognize malicious activities by inspecting audit files, access control procedures and users' credentials. This enables network traffic to be monitored in order to discover denial of service (DoS) and distributed DoS (DDoS) attacks from the network side [34], as suggested in this study.

### *B. Extant Literature*

There has been a number of cloud-based IDS designed for cloud computing environment proposed in the literature [35]. For instance, Nascimento and Correia [16] designed an IDS to protect complex web applications, with their results showing that deploying an IDS in the application layer could be effective. This is because it eases the detection of application attacks via their corresponding running application codes. However, the authors did not provide an effective way of deploying their system in a real-world cloud computing environment. Tupakula *et al.* [34] suggested a hypervisor technique for securing the infrastructure layer from several attack types. Despite the potential to enhance the reliability and accessibility of the system, it could not protect the system if its infrastructure is compromised.

Identifying a suitable collective IDS architecture to be implemented in a cloud computing environment is always a difficult task because of its heterogeneous nature and other characteristics and requirements. Wang *et al.* [36] proposed a collaborative IDS with a central management method to facilitate prompt and precise detection. Vieira *et al.* [37] suggested a grid and cloud computing-based IDS based for recognizing specific attacks. Dhage and Meshram [38] introduced an IDS for cloud computing users, where one controller manages the IDS by applying knowledge-based and artificial neural network (ANN) algorithms. However, the limitations are lack of sensitivity and scalability.

A number of other research studies [2], [39]–[43] have focused on deploying an IDS as software-as-a-service (SaaS). For example, Shelke *et al.* [2] suggested a multi-threaded distributed IDS technique for processing and protecting large-scale cloud-based networks. Zarrabi and Zarrabi [39] proposed a host and network IDS to identify suspicious activities in the cloud. Alharkan and Martin [40] proposed an IDS as a service for a cloud system to detecting cloud attacks. Nikolai and Wang [42] proposed a hypervisor-based cloud IDS. However, these systems are not scalable or sufficiently robust to identify distributed attacks as such attacks operate independently and require long processing times.

Security controls, such as Cloudlet [4] and IOx [44], have also been designed to operate at fog computing nodes. Cloudlet consists of three layers, where the bottom is Linux and a data cache from the cloud, the middle visualization has some cloud software, and the top programs are separated by various VMs. IOx is a Cisco router which operates by hosting programs in a guest OS executing on a hypervisor on the hardware of the grid router. For instance, Shi *et al.* [45] proposed a cloudlet-based security framework for identifying malicious activities of cloud systems to protect communications among the cloudlet, mobile devices and cloud. Yaseen *et al.* [46] developed a technique that provided a global capability to monitor sensors and detect intrusive events using the infrastructure of fog computing. Sandhu *et al.* [47] suggested a framework that includes a Markov model, IDS and virtual honeypot device, for identifying suspicious edge devices in a fog computing environment.

There have also been attempts to use adversarial machine learning to discover vulnerabilities of learning algorithms and mitigate such vulnerabilities. In [48], for example, it was demonstrated that machine learning models can be poisoned using well-crafted APT strategies by targeting a specific vulnerability such as adjusting learning parameters. Huang *et al.* [26] categorized attacks against machine learning models based on their vulnerabilities. Xiao *et al.* [49] studied the loopholes of regularization and feature selection models against data poisoning attacks. A technique to identify optimal poisoning attacks using an outlier detection method was also presented. However, in this study, we determine the impact of data poisoning attacks in high dimension and dynamic data using mixture models-based ADS.

## III. ODM-ADS Mechanism

The proposed ODM-ADS based adversarial learning mechanism is designed by estimating the densities using the DMM [19] for each network instance and detecting abnormal instances located outside the normal boundary [20], [50]. For

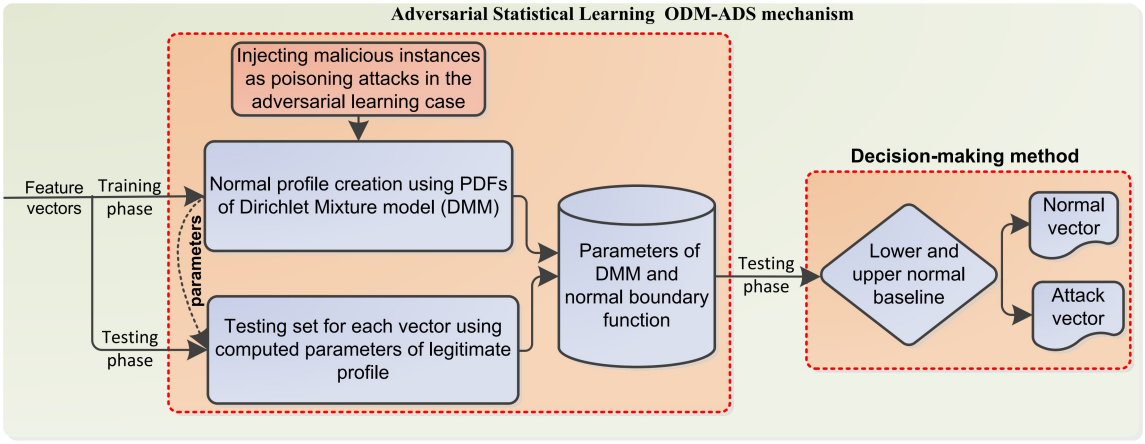


Fig. 1. Proposed adversarial statistical learning ODM-ADS mechanism.

determining the adversarial learning model against poisoning attacks, the mechanism is trained and validated in two cases, namely: with purely normal instances and with injecting some malicious instances, to examine to what extent the model is robust by comparing the two cases. The aim of the mechanism is to develop an autonomous and lightweight ADS for modeling the network big data in dynamic networks along with considering adversarial behaviors. The ODM-ADS is autonomous (automatically adapt the baseline for identifying abnormal instances) and lightweight (internal process requires only some statistical parameters of the DMM and normal boundaries to be estimated, which can be easily achieved in real-time processing).

The mechanism is developed based on the concept of learning theory [51], and includes training, testing and decision-making stages, as shown in Figure 1. In the training stage, the DMM's parameters are computed using: 1) only normal observations and 2) malicious and normal observations as an adversarial case, for building a profile. The same parameters of the profile are used to compute the densities of the testing observations. In the decision-making method, a normal boundary function calculated from the profile is proposed as a baseline, with any variation from it considered an anomaly.

The main reasons for using the DMM are that it can both model non-normal data and specify the legitimate boundary accurately. According to the literature, if data does not follow a Gaussian distribution, using mixture models such as DMM is preferable [15], [19], [23], [50]. In [15], we revealed that the features of network data, such as in the datasets of NSLKDD and UNSW-NB15, cannot precisely follow a normal distribution and are not linearly represented because they do not model its symmetric and unbounded border (i.e., $]-\infty,\infty[$) while they are located in a semi-bounded $[0, N]$ range, where $N$ is an integer or real number.

According to [52]–[54], the DMM accurately identify the bounds of network data as it contains a set of probability distributions and is more suitable for fitting time-series data (e.g., network data). Also, its features enable samples to be represented in clustered distributions [52]. Therefore, we apply this model to properly fit network data with the normal boundary as a baseline outside of which any vector is identified as an anomaly, as detailed in the following three subsections.

### A. Finite DMM

Since a finite mixture model is a convex collection of many Probability Density Functions (PDFs) it is a resilient probabilistic model for multivariate data [15], [55]. A finite mixture of Dirichlet distributions including $M$ components is computed by [52] and [56]

$$p(X|\pi,\alpha)=\sum_{l=1}^{M}\pi_l Dir(X|\alpha_i) \tag{1}$$

where $\pi = (\pi_1,\ldots,\pi_M)$ indicates the mixing coefficients, which are usually positive, with their summation 1, $\sum_{l=1}^{M}\pi_l$,$\alpha = (\alpha_1,\ldots,\alpha_M)$, and $Dir(X|\alpha_j)$ refers to the Dirichlet distribution of component $j$ with its own positive parameters $(\alpha=(\alpha_{j1},\ldots,\alpha_{jD})$ computed by

$$Dir(X|\alpha_j)=\frac{\Gamma(\sum_{d=1}^{D}\alpha_{jd})}{\prod_{d=1}^{D}\Gamma(\alpha_{jd})}\prod_{d=1}^{D}X_d^{\alpha_{jd}-1} \tag{2}$$

where $X=(X_1,\ldots,X_D)$, $D$ stands for the dimensions of $X$ and $\sum_{d=1}^{D}x_d=1, 0\le X_d\ge 1$, for $d=1,\ldots,D$. A Dirichlet distribution could be utilized as a posterior distribution to model the data instead of using its as a prior distribution for multinomial data.

Let assume there be a set of $N$ $iid$ observations $(X=\{X_1,\ldots,X_N\})$ that can be fitted using the mixture distribution as in equation (2), the probability function of the DMM is declared as

$$p(X|\pi,\alpha)=\prod_{l=1}^{N}\{\sum_{j=1}^{M}\Pi_j Dir(X_l|\alpha_j)\} \tag{3}$$

As the finite mixture model in equation (2) is a latent flexible one, for each vector $(X_i)$, a $M$-dimensional arbitrary vector $(Z_j=\{Z,\ldots,Z_{jM}\})$ is defined, where $Z_{jd}\in\{0,1\}$, $\sum_{i=1}^{M}$ and $Z_{jd}=1$ if $X_j$ follows component $j$, else 0. The latent variables $(Z=\{Z_1,\ldots,Z_N\})$, which are concealed in

the model, and their conditional distribution are given by the mixing coefficients ($\pi$) and calculated by

$$p(Z|\pi) = \prod_{l=1}^{N}\prod_{j=1}^{M} \pi_j^{Z_{lj}} \tag{4}$$

The likelihood with hidden variables that represents the conditional distribution of a dataset ($dt$) is formulated as

$$p(X|\pi,\alpha) = \prod_{l=1}^{N}\prod_{j=1}^{M} Dir(X_l|\alpha_j) \tag{5}$$

Given a dataset ($dt$), a significant challenge is computing the parameters of the mixture models and specifying their numbers of components ($M$). Firstly, to compute the parameters of the finite DMM, we use the variational inference provided in [52] and [53] and briefly discussed in the following subsection. Secondly, based on the principal components of the Principal Component Analysis (PCA) technique [15], [57], we use the number of components that generates the highest variations in features to determine the differences between normal and malicious feature vectors [15].

*1) Parameter Estimation for DDM:* Variational learning is used to estimate the parameters of the DMM [52], [53]. Firstly, the conjugate priors over parameters $\alpha$, $\beta$, $\lambda$ and $\tau$ model the prior data with the same distribution as the posterior, with the Beta parameters statistically independently adapted to these parameters, as provided in [54], to estimate the conjugate prior, where $p(\alpha) = G(\alpha|u,v)$, $p(\beta) = G(\beta|p,q)$, $p(\lambda) = G(\lambda|g,h)$ and $p(\tau) = G(\tau|s,t)$. $G(\cdot)$ denotes the gamma distribution and is computed via $G(x|a,b) = \frac{b^a}{\Gamma(a)}x^{a-1}e^{-bx}$.

To simplify the representation, let $\Theta = \{Z, W, \varphi, \alpha, \beta, \lambda, \tau\}$ be a set of non-observed arbitrary variables and $\Lambda = \{\pi, \eta, \varepsilon\}$ a set of parameters. The main target is to optimize the values of $\Lambda$ by maximizing the marginal likelihood ($p(X|\Lambda)$) to obtain the best values for modeling network data using the DMM. As this marginalization is inflexible, the variational model [52] is then applied to find a tractable lower boundary for $p(X|\Lambda)$. By using Jensen's inequality, the lower boundary ($L$) of the logarithm of $p(X|\Lambda)$ is calculated by

$$\ln\, p(X|\Lambda) \geq \int Q(\Theta)\ln\frac{p(X,\Theta|\Lambda)}{Q(\Theta)}d\Theta = \mathcal{L}(Q) \tag{6}$$

where $Q(\Theta)$ is an estimate of the accurate posterior distribution ($p(\Theta|X,\pi)$), with the factorization assumptions for limiting its form defined as

$$Q(\Theta) = Q(\mathbb{Z})Q(\overrightarrow{\phi})Q(W)Q(\overrightarrow{\alpha})Q(\overrightarrow{\beta})Q(\overrightarrow{\lambda})Q(\overrightarrow{\tau}) \tag{7}$$

To maximize the lower boundary ($L(Q)$), a variational optimization of it for each distribution ($Q_i(\Theta_i)$) is used by defining a certain factor ($Q_s(\Theta_s)$) and computing its optimal solution by

$$Q_s(\Theta_s) = \frac{\exp\langle\ln p(X,\Theta)\rangle_{i\neq s}}{\int \exp\langle\ln p(X,\Theta)\rangle_{i\neq s}\, d\Theta} \tag{8}$$

The variational lower boundary ($L(Q)$) can be iteratively appraised to observe the convergence and measure the accuracy of the variational learning method. In a re-estimation, in order to obtain the best true marginal log-likelihood of $L(Q)$, the value of the lower boundary should never decrease. Then, the model parameters ($\Lambda$) are computed by maximizing the boundary based on the estimated parameters $\pi$, $\eta$ and $\epsilon$. When the derivative of the lower boundary with respect to $\pi_j$, $\eta_{jlk}$ and $\epsilon_{jl}$ is set equal to zero, the following formulas can be obtained.

$$\pi_j = \frac{1}{N}\sum_{i=1}^{N} r_{ij}, \quad \eta_{jlk} = \frac{1}{N}\sum_{i=1}^{N} m_{ij}, \quad \epsilon_{jl} = \frac{1}{N}\sum_{i=1}^{N} f_{ijl} \tag{9}$$

Because the solution for the variational posterior ($Q$) and value of the lower boundary rely on the values of $\pi$, $\eta$, and $\epsilon$, optimization of the variational model can be addressed by the EM algorithm [52]. In the variational equivalent of the E-step, the results for each variational factor in equations (6) to 9) are optimized. Then, in the variational equivalent of the M-step, the lower boundary ($L(Q)$) is maximize using the current values of $\pi$, $\eta$, and $\epsilon$. In summary, the three main DMM parameters ($\pi, \alpha, Z$) are estimated using the variational learning technique by estimating its boundaries for building an adaptive adversarial learning ADS mechanism for efficiently detecting zero-day attacks, as elaborated below.

### *B. Training Phase of ODM-ADS Mechanism*

The methodology for correctly applying anomaly detection depends mainly on training normal data in only the training phase while constructing a legitimate profile and treating any variation from it as an attack observation in the testing phase. In addition, the adversarial learning is utilized along with this methodology by injecting some malicious instances in the training phase for measuring the trustworthiness of the proposed ODM-ADS mechanism. This can be achieved by considering normal ($O_{1:n}^{normal}$) and attack ($O_{1:l}^{attack}$) observations as the entire observations are normal. Each observation includes a set of features $F$ where $O_{1:n}^{normal} = \{f_1, f_2, \ldots, f_D\}^{normal}$ and $O_{1:l}^{attack} = \{f_1, f_2, \ldots, f_D\}^{attack}$ that produce the profile including the statistical characteristics of the observations.

There are two cases: 1) training with normal data, and training with normal and attack data dealt as normal, applied to measure the strength of the model. Different attack samples are picked from the datasets using the random sampling method, where the number of attack samples is lower than the number of normal samples in order to solve the over-fitting problem and satisfy the generalization of the model. In this sense, the mechanism can learn the patterns of normal observations using the DMM with lower attack instances that could not degrade the mechanism's performance while training a huge number of normal instances.

In the methodology, the profile contains the estimated parameters ($\pi, \alpha, Z$) of the DMM ($Dir(X|\pi,\alpha,Z)$) for estimating the PDF for each feature vector in the training set. The process of constructing a profile ($prof$), with the parameters ($\pi, \alpha, Z$) of the DMM estimated for the entire observations using the equations in Subsection III-A.1, is described in Algorithm 1. Then, the PDFs of the network features ($f_{1:D}$) are computed using equation (5) based on the estimated parameters in order to fuse important features in a representative probability value for each vector. This assists in perfectly fitting network data

into a profile in which its boundaries can be automatically specified. We propose a boundary function that can dynamically specify the lower and upper boundaries of the legitimate PDFs, respectively, as

$$lower^{normal} = \mu(pdf^{normal}) - (w * \sigma(pdf^{normal})) \quad (10)$$

$$upper^{normal} = \mu(pdf^{normal}) + (w * \sigma(pdf^{normal})) \quad (11)$$

where $\mu$ and $\sigma$ are the mean and standard deviation, respectively, and $w$ a specific range of [1.5, 3] [50], [58] that indicates the lowest and highest variations of the normal data points, as explained below.

**Algorithm 1** Training Phase in ODM-ADS Mechanism

**Input**: normal observations ($O^{normal}_{1:n}$)/ normal and attack observations as normal ($O^{normal}_{1:n}$) in the case of adversarial learning
**Output**: normal profile ($prof$)
1: **for** each row i in ($O^{normal}_{1:n}$)
2: estimate the parameters ($\pi_i, \alpha_i, Z_i$) of the DMM discussed in Subsection III-A.1
3: compute the PDFs using equation (5) based on step 2
4: **end for**
5: compute $lower^{normal}$ and $upper^{normal}$ using equations (24) and (25)
6: $prof \leftarrow(\pi, \alpha_i, Z_i, lower^{normal}, upper^{normal})$
7: **return** $prof$

### C. Testing Phase in ODM-ADS Mechanism

The Dirichlet PDF ($PDF^{testing}$) of each testing observation ($O^{testing}$) is calculated based on the same parameters computed for the legitimate profile ($prof$). In Algorithm 2, the procedures in the testing stage for recognizing the PDFs of suspicious observations is discussed. Step 1 declares estimating the PDF of every testing observation by applying the same parameters ($\pi, \alpha, Z$) of the training phase.

**Algorithm 2** Steps of Testing and Attack Detection in ODM-ADS Mechanism

**input** : observed instance ($O^{testing}$), $prof$
**output** : normal or attack instance
1: estimate the $PDF^{testing}$ using equation (5) with parameters ($\pi_i, \alpha_i, Z_i$)
2: **if** ($PDF^{testing} \geq lower^{normal}$ || $PDF^{testing} \leq upper^{normal}$)
3: **return** normal
4: **else**
5: **return** attack
6: **end if**

Steps 2 to 6 demonstrate the fundamental steps in the decision-making method. More specifically, the lower and upper boundaries of the profile are calculated to discover abnormal observations in the testing set ($O^{testing}$) by considering the observations below the lower or above the upper baselines, where $w$ refers to interval values between 1.5 and 3. This interval mathematically proves that any data from the same distribution can be varied in this particular range with any data points located outside this range considered outliers.

In other words, the fewest and most dispersion of data points fitted under a probability distribution can vary from those expected ($\mu$) in ranges of (–[1.5, 3] $* \sigma$) as a lower boundary and (+[1.5, 3] $* \sigma$) as an upper boundary [50], [58]. Similarly, in network anomaly detection, we consider that the detection decision relies on handling any $PDF^{testing}$ located outside this range as an anomaly instance, otherwise a normal vector, which produces the promising results discussed in Section V.

## IV. Proposed ODM-ADS Framework as SaaS

In this section, we explain the proposed framework for designing an autonomous and lightweight anomaly detection scheme that can reliably discover malicious observations dynamic and large-scale networks. As depicted in Figure 2, it is deployed at the fog side to address the limitations of the cloud and detect abnormal behaviors using two steps: a smart distributed data module as SaaS; and an ADS as SaaS. The former contains a sensor for sniffing network features and recording them in a distributed database. It is designed to be an intelligent and shareable service for all the ADSs connected at different choke points (i.e., ingress routers and switches) in order to facilitate the processing and management of big data on the fog nodes.

In more detail, the smartly distributed data module sniffs and collects network data from different fog nodes at the destination nodes of ingress routers for handling at the edge of the network to improve computational resources, reduce network overheads and support mobility. For example, the architecture of the UNSW-NB15 dataset and its connection to the proposed ADS as SaaS is presented in Figure 3. In it, the IXIA traffic generator is configured to simulate contemporary network packets from different IoT devices. The tcpdump is used to extract the packets and log them in pcap files, and then the tools of Argus and BRO are applied for extracting network features. The MySQL Cluster CGE technology [59] can be used in the smartly distributed data module as SaaS to store and handle the data of features because it has an extremely scalable and real-time database. The important features selected using the PCA technique are passed to the proposed ADS approach for detecting anomalous behaviors in fog nodes which provides a great deal of protection at the network edges as the choke points are concurrently monitored and analysed.

The framework continually monitors the destination nodes the fog to considerably improve the use of computational resources and efficiency of attack detection. The collaborative ADS as SaaS module involves the main functions of the data pre-processing module and ODM-ADS mechanism to be implemented at each node to handle the network big data of computing systems by distributing it as a service at each node. The data pre-processing module filters and selects important features from network traffic using the PCA technique, as explained in [15], while the ODM-ADS mechanism recognizes abnormal observations in the dynamic and large-scale networks.

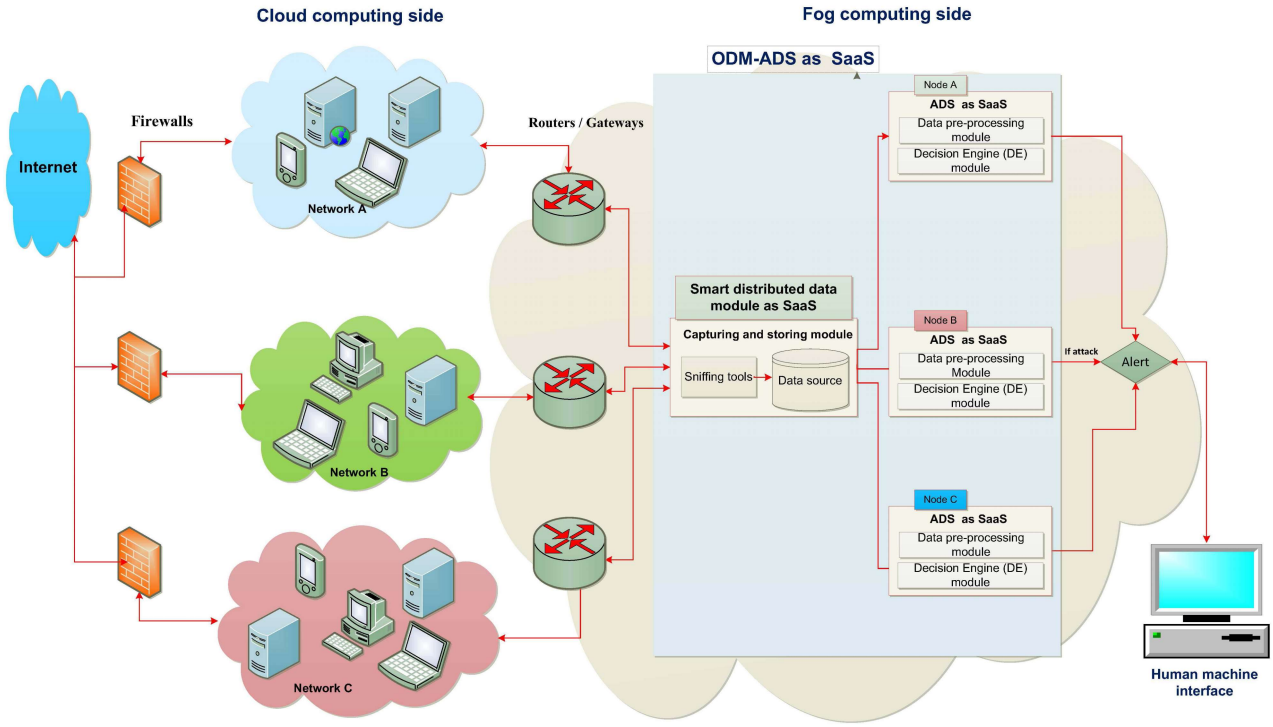


Fig. 2. Proposed ODM-ADS framework as SaaS in fog.

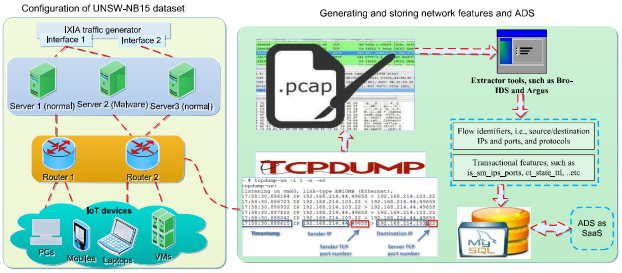


Fig. 3. Architecture of smartly distributed data (using UNSW-NB15 dataset as example) and its connection with ADS as SaaS for fog and cloud sides.

This framework is deployed for the three nodes depicted in Figure 2, namely A, B and C, to be executed for fog and cloud computing systems. Unlike traditional IDSs, the ADS as SaaS is deployed at each network node and each instance is connected to the shared module to capture and record network flows. This process collects the feature values of network traffic in a specific time interval which makes it much easier to pass the processed data to the DE module for each network node.

The deployment of the ODM-ADS mechanism on the fog side can address the challenges of low latency, mobility support, location awareness and geo-distribution for several reasons. First and foremost, it can be practically executed by monitoring network traffic at the choke points, such as ingress routers, switches and gateways, which can reduce the network traffic's overhead and also help to aggregate only the relevant flows that should be inspected to identify abnormal activities [3], [15]. Secondly, network data exchanges between the cloud and fog sides are monitored and inspected at the fog's devices and applications at which nodes network activities can be clearly tracked.

TABLE I
COMPARISONS OF LATENCY AND BANDWIDTH

| Systems | Latency (MS) | Bandwidth (Mbps) | |
|---|---|---|---|
| | RTT | Uplink | Downlink |
| Cloud | 18.898 | 82.687 | 102.813 |
| Fog | 1.409 | 1.643 | 1.725 |

Finally, since the ODM-ADS mechanism is developed based on estimating some statistical measures that can be simply computed in real time, examining either normal or anomalous network flows at network edges (fog) is much faster than processing at network cores (cloud). Table I presents comparisons of the latencies measured using the RTT (round-trip time), computed uplink and downlink bandwidths of CISCO's IOx fog and cloud to demonstrate their performances. It is clear that the fog computation has greater benefits than the cloud in terms of latency and bandwidth.

## V. EVALUATION

### A. Pre-Processing Phase

The ODM-ADS mechanism is developed using the 'R programming language' on Linux Ubuntu 14.04 with 16 GB RAM and an i7 CPU processor. In the pre-processing phase of the

framework, we select the NSL-KDD [21], [22] dataset because it is widely used to assess NIDSs, and the UNSW-NB15 dataset [23]–[25] as it contains a broad variety of contemporary normal and abnormal network observations which enable rational comparisons of the performances of the proposed ODM-ADS framework and recent peer techniques.

The NSL-KDD dataset is an enhanced version of the KDD CUP 99 dataset that tackles some of the latter's issues, for example, it eliminates duplicated vectors in the training and testing sets to reduce the possibility of any classification technique being biased towards the vectors with the highest frequencies. Each vector consists of 41 features and the labels of five types of classes, one normal and four attacks (i.e., DoS, Probe, U2R and R2L). The UNSW-NB15 dataset includes a hybrid of recent legitimate and attack vectors. The volume of each of its network packets is approximately 100 Gigabytes extracted from 2,540,044 vectors and recorded in four CSV files. Each vector consists of 47 features and the labels of ten types of classes, one normal and nine attacks (i.e., Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Fuzzers for anomalous activity, Shellcode and Worms).

Many experiments are conducted on both datasets to estimate the suggested framework's performance using the criteria of accuracy, DR and FPR, as explained below, which rely on the four terms True Positive (TP), True Negative (TN), False Negative (FN) and False Positive (FP). TP and TN are the numbers of actual malicious vectors identified as legitimate and malicious, respectively, and FP and FN are the numbers of actual legitimate vectors incorrectly identified as legitimate and malicious, respectively [15].

- The accuracy indicates the proportions of entirely legitimate and malicious vectors successfully predicted, that is,

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{12}$$

- The DR is the proportion of correctly identified malicious vectors, that is,

$$DR = \frac{TP}{TP + FN} \tag{13}$$

- The FPR is the proportion of incorrectly identified malicious vectors, that is,

$$FPR = \frac{FP}{FP + TN} \tag{14}$$

We selected arbitrary observations from the 'full' NSL-KDD dataset and CSV files of the UNSW-NB15 dataset, with diverse sample sizes of between 150,000 and 350,000 (s). For estimating the robustness of the mechanism's learning process, we train and validate the mechanism using two cases: only normal data, and normal and attack data as normal. Legitimate samples are chosen to be about 60-75% of the whole relevant sample size in the first case, while the attack samples are about 10-15% of the whole sample size in the second case. The performance of this technique is assessed using a 10-fold cross-validation [15], [30], [31] of the sample sizes to inspect their impacts with no bias towards some samples.

The ODM-ADS mechanism is assessed using 15 features from both datasets selected using the PCA technique published in [15]. The largest number of features with higher variances is chosen to reveal that each feature can affect the performance of the DE technique which is aimed at correctly applying the learning theory, whereby the selected features do not depend on each other but on the predictor (i.e., class label), to improve the efficacy of establishing a machine-learning technique.

### *B. Role of Statistical Analysis in Developing ODM-ADS Mechanism*

Statistical analysis has a great effect regarding the design of the ODM-ADS mechanism in terms of understanding network data patterns by showing to what extent suspicious vectors are dissimilar from normal ones, with two statistical measures, density [60] and correntropy [61], applied to network data to determine the variances between these vectors.

The density probabilities of normal and suspicious vectors are computed using some samples from the NSL-KDD and UNSW-NB15 datasets to reveal to what extent they vary, as shown in Figure 4. In both datasets, those of normal observations are somewhat different from those of suspicious ones and it is clear that the proposed ODM-ADS mechanism can adequately identify malicious vectors due to these differences.

Also, the correntropy plots in Figure 5 statistically demonstrate the small differences between legitimate and malicious feature vectors of some instances in both datasets which simplifies the role of the ODM-ADS technique in successfully discovering anomalous vectors. The variations occur because the majority of features in both datasets are established via analysing their potential statistical characteristics, for example, inter-arrival times and packet counts from header network packets. These characteristics lead to identifying clear differences between normal and abnormal vectors, thereby considerably improving the performance of the ODM-ADS mechanism when deployed in real fog and cloud environments.

As the statistical measures prove that the network features cannot be fitted linearly or normally because recent abnormal activities try to mimic normal observations for the design of the proposed ODM-ADS technique. They highlight that can precisely fit such a network with respect to currently dynamic networks. This also indicates that if an attacker injects some poisoning instances in the training phase, the utilization of mixture models, such as the DMM, could identify these instances while using a huge number of legitimate instances in the training phase.

### *C. Performance Evaluation of ODM-ADS Mechanism*

The performances of the ODM-ADS mechanism are evaluated using the two cases (i.e., only normal data, as well as normal and attack as normal) in terms of DR, accuracy and FPR. In the two datasets using the first case listed in Table II (a), the DR and accuracy are gradually improved while the FPR slightly decreased to 0.21%, along with increasing the w value to 3. In the second case of the adversarial learning process described in Table II (b), we observe that there is a

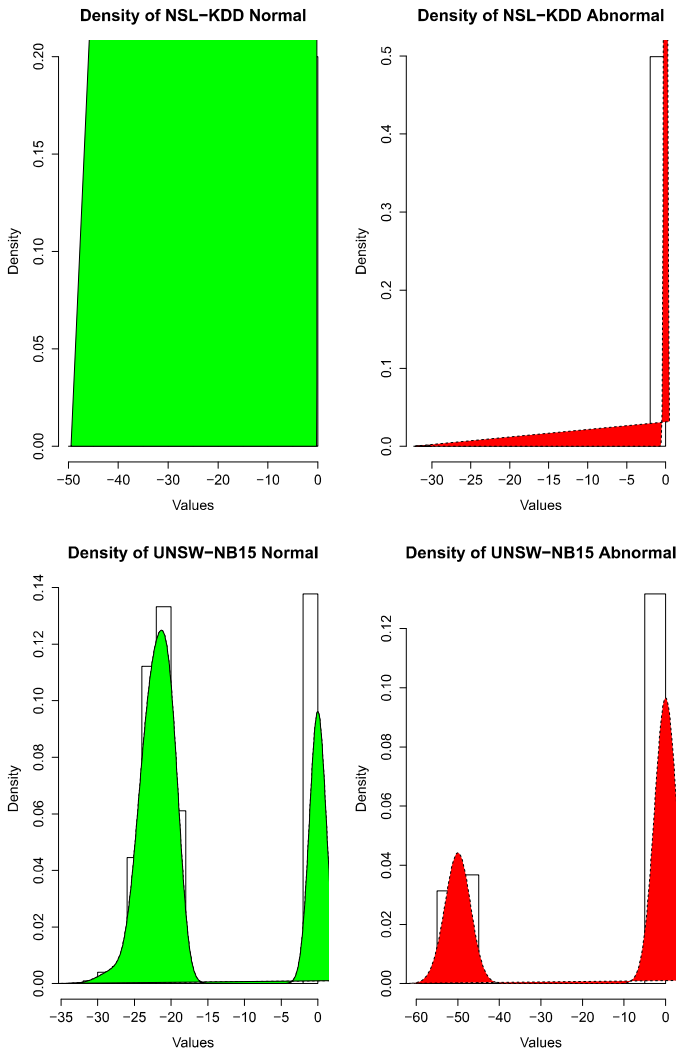


Fig. 4. Density probabilities of some normal and abnormal instances in both datasets.

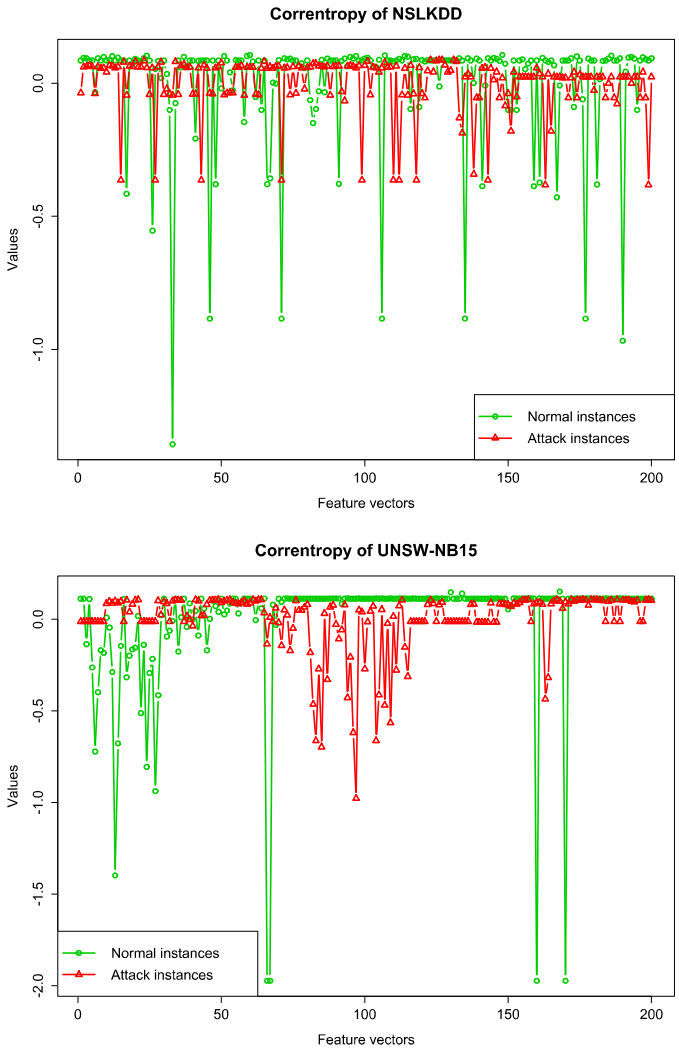


Fig. 5. Correntropy of some feature vectors using both datasets.

TABLE II

EVALUATION OF OVERALL PERFORMANCES OF ODM-ADS TECHNIQUE

| Only normal data in the training phase | | | | | | |
|---|---|---|---|---|---|---|
| | NSL-KDD dataset | | | UNSW-NB15 dataset | | |
| w value | DR (%) | Accuracy (%) | FPR (%) | DR (%) | Accuracy (%) | FPR (%) |
| 1.5 | 95.54 | 96.27 | 0.46 | 91.38 | 92.87 | 6.93 |
| 2 | 96.75 | 97.67 | 0.37 | 93.46 | 94.65 | 5.35 |
| 2.5 | 98.38 | 98.91 | 0.29 | 95.56 | 95.82 | 4.72 |
| 3 | 99.89 | 99.92 | 0.21 | 97.43 | 98.74 | 2.83 |

(a)

| Normal and attack data as normal in the training phase | | | | | | |
|---|---|---|---|---|---|---|
| | NSL-KDD dataset | | | UNSW-NB15 dataset | | |
| w value | DR (%) | Accuracy (%) | FPR (%) | DR (%) | Accuracy (%) | FPR (%) |
| 1.5 | 94.5 | 95.45 | 0.57 | 93.52 | 93.46 | 7.05 |
| 2 | 96.38 | 96.58 | 0.46 | 93.58 | 94.25 | 6.85 |
| 2.5 | 97.67 | 98.32 | 0.38 | 94.64 | 95.93 | 5.69 |
| 3 | 98.25 | 98.59 | 0.31 | 95.46 | 96.82 | 3.91 |

(b)

slight difference in the terms of accuracy, DR and FPR that do not degrade the mechanism's performance.

This is because that the model was trained by using a huge number of legitimate data in the training set. As the normal data is the dominated class, the attack data slightly change the statistical parameters of the DMM. We selected the highest variation (i.e., w = 3) in the training phase that produces the highest performances, so the adapted parameters are still reliable for generating the baseline of the training phase. It is worth mentioning that the mechanism can identify zero-day attacks using the two cases with lower than 1% DR as a difference, therefore, the mechanism is trustworthy against poisoning attacks. We use the outcomes of the first case in the following discussions and comparisons with the other techniques. To show the capability of the proposed mechanism for identifying normal and attack types, Figure 6 presents the detection rates of the types for the two datasets with different w values. It is obvious that the detection rates increasingly improve with gradual increases in the w values from 1.5 to 3.

Some malicious observations, in particular, Shellcode, Fuzzers, Reconnaissance and Backdoor in the UNSW-NB15 dataset, do not attain the highest DRs with gradual increases in the w values from 1.5 to 3 while those of the other types of attacks, Generic, DoS, Exploits and Worms, are better

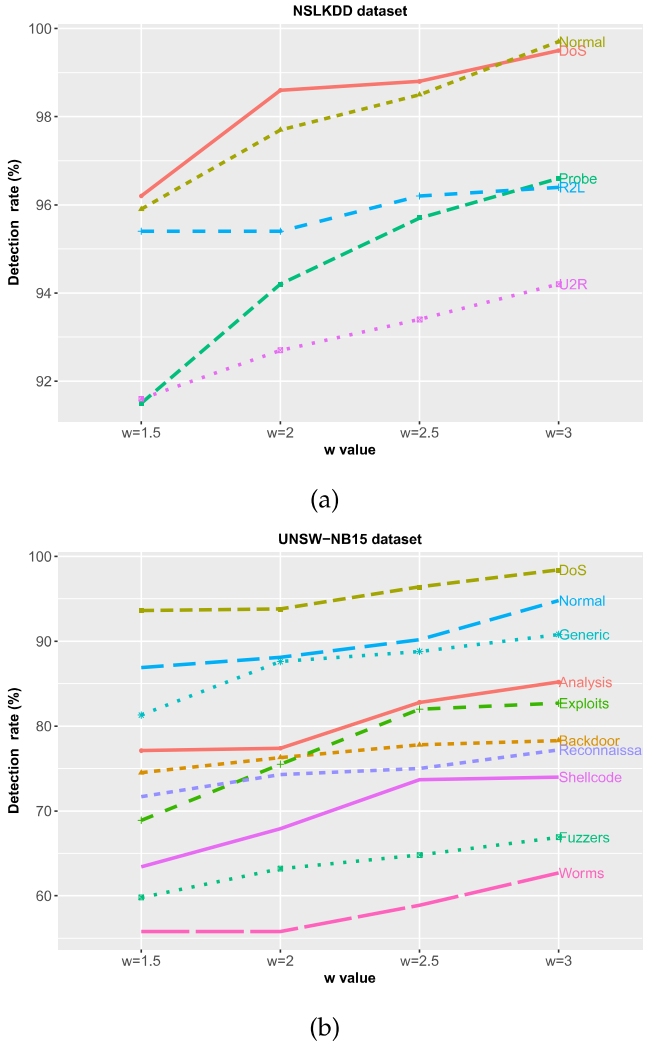


Fig. 6. Comparisons of DRs (%) obtained from ODM-ADS mechanism using both datasets with different w values.

because of the small differences between these malicious and normal observations. It can be observed that the probability densities of the variances in the features for these instances from the UNSW-NB15 dataset are sometimes close, especially those of recent sophisticated attacks that try to mimic normal observations. Therefore, there is some overlap while running the threshold of the proposed decision-making method.

There are major reasons that the ODM-ADS mechanism can efficiently detect unknown attacks. The DMM can accurately fit the bounds the selected features because its modeling includes multiple probability distributions, including priors, likelihoods and posteriors, of the network data in order to precisely calculate the probability density of each feature vector. The normal boundary can effectively specify the baseline boundaries between normal and anomalous observations.

### D. Comparative Analysis

The performances of the ODM-ADS mechanism and seven NIDS techniques, namely, the Multivariate Correlation Analysis (MCA) [30], Computer Vision Technique (CVT) [62], Triangle Area Nearest Neighbors (TANN) [63], Artificial Immune System (AIS) [64], Euclidean Distance Map (EDM) [31], Filter-based Support Vector Machine (FSVM) [65] and Geometric Area Analysis (GAA-ADS) [15], are compared. As can be seen in Table III, the experimental results clearly demonstrate the superiority of the ODM-ADS technique in terms of the detection rate and false positive rate on the NSL-KDD dataset.

TABLE III

COMPARISON OF PERFORMANCES OF SEVEN ADS TECHNIQUES USING NEW ODM-ADS ON NSL-KDD DATASET

| Technique | DR | FPR |
|---|---|---|
| EDM [31] | 94.36% | 7.28% |
| MCA [30] | 96.38% | 4.97% |
| TANN [63] | 91.25% | 9.46% |
| CVT [62] | 95.36% | 5.10% |
| AIS [64] | 90.27% | 9.93% |
| FSVM [65] | 92.43% | 8.82% |
| GAA-ADS (original features) [15] | 98.26% | 0.57% |
| GAA-ADS (components) [15] | 98.75% | 0.43% |
| Our ODM-ADS | 99.89% | 0.21% |

The first four techniques were developed to identify only DoS attacks, for which they attain higher DRs, not U2R, U2L and Probe malicious observations. Because they rely on computing the distances and correlations between legitimate and suspicious observations, as different attacks, specifically stealthy and spy intrusion activities [30], [62], dramatically mimic legitimate instances, they overlap the legitimate profile and reduce the DR.

The FSVM and AIS techniques were developed to learn legitimate and malicious instances in the training phase based on the rule-based principle. They usually require a massive number of instances to successfully learn diverse patterns that are difficult to find in real network environments. The assessments reveal that their DRs are superior for DoS and Probe attacks, of which there are sufficient observations, but worse for less frequent intrusive activities, such as U2R and U2L attacks.

The GAA-ADS technique can perform better than the above mechanisms for different kinds of attacks as it precisely estimates the geometric area of the vectors. Finally, the ODM-ADS mechanism demonstrates its superiority over the other approaches for identifying attack types with different w values because the DMM can perfectly fit the boundaries of each network feature since its modeling involves some probability distributions for accurately estimating the probability densities of each feature vector. Furthermore, the baseline can exactly specify the boundaries between legitimate and suspicious instances.

### E. Complexity and Time Cost of ODM-ADS Mechanism

The complexity and processing time of the ODM-ADS mechanism are analyzed and compared with those of the other approaches in order to demonstrate its effective and reliable performance, with the results presented in Table IV. This technique has three main steps: 1) calculating the complexity of the PCA which is $O(ND \times min(N, D))$; 2) computing

TABLE IV
COMPARISONS OF COMPLEXITY LEVELS OF ADS MECHANISMS

| Technique | Complexity of training phase | Complexity of testing phase |
|---|---|---|
| MCA [30] | $O(ND^2)$ | $O(ND^4)$ |
| EDM [31] | $O(ND^2)$ | $O(ND^4)$ |
| TANN [63] | $O(NDL^2)$ | $O(N^2DL^2)$ |
| CVT [62] | $O((2ND^2)$ | $O(ND^4))$ |
| GAA-ADS [15] | $O((ND \times min(N, D)) + ND^3)$ | $O(ND + 1)$ |
| ODM-ADS | $O((ND \times min(N, D)) + ND^3 + N)$ | $O(ND))$ |

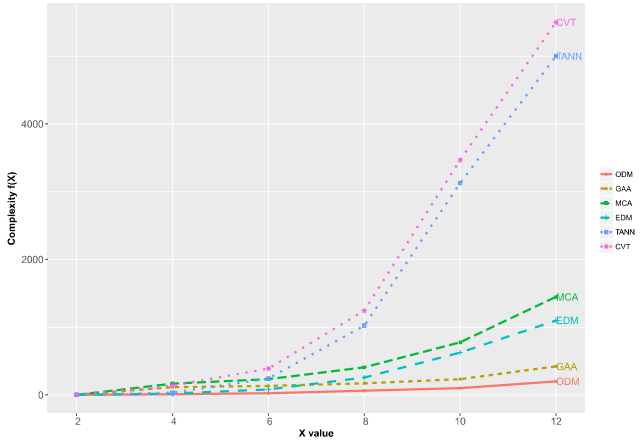


Fig. 7. Comparison of complexity levels of ODM-ADS and other techniques.

the ODM parameters ($Dir(\pi, \alpha, Z)$); and 3) calculating the lower and upper boundary of the normal profile. The three big $O$ notations are integrated for these steps to calculate the technique's total complexity in terms of handling $N$ network instances, each with $D$ features.

On the one hand, in the training phase, the ($Dir(\pi, \alpha, Z)$) parameters take $O(ND^3)$, with the $D$ features within their samples approximated while the normal boundary generates $O(N)$. Similar to the GAA-ADS technique, the testing phase takes $O(ND)$ due to applying the concept of 'record by record' detection. As a consequence, for all records, the overall complexity of the ODM-ADS technique is $O((ND \times min(N, D)) + ND^3 + N + ND)$ and, as the $(ND^3)$ term becomes greater than the others, its final complexity is $O(ND^3)$. Similar to the GAA-ADS technique, the $D$ features are $iid$ in each record and performed simultaneously, with the overall complexity $O(1)$.

Figure 7 presents a comparison of the levels of complexity of the four ADS and new ODM-ADS technique using the equation of the big $O$ notation ($f(X) = O(g(X))$), where $g(X)$ denotes the estimated overall complexity and $X$ integer numbers. Since the number of observations increases, the ODM-ADS technique runs more quickly than the others.

An examination of the time costs indicates that the ODM-ADS technique has a faster processing speed than the others. It can process almost 23,890 observations per second whereas the GAA-ADS, MCA, CVT and EDM techniques can handle approximately 23,696, 23,092, 19,267 and 12,044, respectively. Overall, the ODM-ADS can run, on average, 1.04%, 1.09%, 2.21% and 10.34% faster than the GAA-ADS, MCA, CVT and EDM mechanisms, respectively.

### *F. Advantages and Disadvantages of ODM-ADS Mechanism*

The ODM-ADS mechanism has many advantages. Firstly, it is simple to implement in large-scale networks for detecting suspicious activities in real time as the preparation of its training and testing phases relies on only computing some parameters to construct a profile, along with injecting some malicious instances in the training phase. Because the DE method applies the boundary function as a baseline, it can define the class label of each observation without depending on other observations. Also, it is very easy to update the parameters of the profile with respect to selecting the best baseline that can improve the performance of the mechanism in terms of a low processing time, high DR and accuracy, and low FPR.

To ensure the best performance of this mechanism, a huge amount of pure legitimate instances is required to produce the highest DRs and lowest FARs. While this mechanism is designed to process binary classifications (i.e., normal or abnormal), in future, it will be enhanced to detect malicious types, such as DoS and Backdoor. Also, in order to determine the obvious differences between normal and abnormal instances, the PCA technique is used to decrease the number of network features with the highest variations among their observations to considerably improve the performance of the proposed ODM-ADS mechanism.

## VI. CONCLUDING REMARKS

This paper presented an adversarial learning ODM-ADS mechanism for discovering suspicious instances and estimating the robustness of statistical learning algorithms against poisoning attacks. To evaluate the mechanism's utility, two cases (normal data, as well as normal and attack data labeled as normal) were applied. The findings revealed that the mechanism can effectively identify zero-day attacks in the two cases from dynamic networks with small differences in the detection accuracy levels. This mechanism was evaluated using two different, independent datasets. A fog-based framework was also designed to mitigate limitations in cloud environment (i.e., lack of low latency, mobility support, location awareness, and geo-distribution).

Future research includes implementing a prototype of the proposed ODM-ADS in a real-world fog-based environment (e.g., smart campus), which will also allow us to more effectively evaluate its scalability and capability to deal with the demands and requirements of the real-world application.

## REFERENCES

[1] A. Castiglione, K.-K. R. Choo, M. Nappi, and S. Ricciardi, "Context aware ubiquitous biometrics in edge of military things," *IEEE Cloud Comput.*, vol. 4, no. 6, pp. 16–20, Nov./Dec. 2017.

[2] M. P. K. Shelke, M. S. Sontakke, and A. D. Gawande, "Intrusion detection system for cloud computing," *Int. J. Sci. Technol. Res.*, vol. 1, no. 4, pp. 67–71, 2012.

[3] Z. Tan *et al.*, "Enhancing big data security with collaborative intrusion detection," *IEEE Cloud Comput.*, vol. 1, no. 3, pp. 27–33, Sep. 2014.

[4] S. Yi, C. Li, and Q. Li, "A survey of fog computing: Concepts, applications and issues," in *Proc. Workshop Mobile Big Data*, 2015, pp. 37–42.

[5] M. Sookhak, A. Gani, M. K. Khan, and R. Buyya, "Dynamic remote data auditing for securing big data storage in cloud computing," *Inf. Sci.*, vol. 380, pp. 101–116, Feb. 2017.

[6] S. Alonso-Monsalve, F. García-Carballeira, and A. Calderón, "Fog computing through public-resource computing and storage," in *Proc. 2nd Int. Conf. Fog Mobile Edge Comput. (FMEC)*, 2017, pp. 81–87.

[7] Y. Tang, D. Chen, L. Wang, A. Y. Zomaya, J. Chen, and H. Liu, "Bayesian tensor factorization for multi-way analysis of multi-dimensional EEG," *Neurocomputing*, vol. 318, pp. 162–174, Nov. 2018.

[8] D. Chen, Y. Hu, L. Wang, A. Y. Zomaya, and X. Li, "H-PARAFAC: Hierarchical parallel factor analysis of multidimensional big data," *IEEE Trans. Parallel Distrib. Syst.*, vol. 28, no. 4, pp. 1091–1104, Apr. 2017.

[9] S. Iqbal *et al.*, "On cloud security attacks: A taxonomy and intrusion detection and prevention as a service," *J. Netw. Comput. Appl.*, vol. 74, pp. 98–120, Oct. 2016.

[10] D. Zissis and D. Lekkas, "Addressing cloud computing security issues," *Future Gener. Comput. Syst.*, vol. 28, no. 3, pp. 583–592, 2012.

[11] N. Moustafa, G. Creech, E. Sitnikova, and M. Keshk, "Collaborative anomaly detection framework for handling big data of cloud computing," in *Proc. Mil. Commun. Inf. Syst. Conf. (MilCIS)*, 2017, pp. 1–6.

[12] J. Chase, D. Niyato, P. Wang, S. Chaisiri, and R. Ko, "A scalable approach to joint cyber insurance and security-as-a-service provisioning in cloud computing," *IEEE Trans. Depend. Sec. Comput.*, to be published.

[13] A. Paudice, L. Muñoz-González, A. Gyorgy, and E. C. Lupu. (2018). "Detection of adversarial training examples in poisoning attacks through anomaly detection." [Online]. Available: https://arxiv.org/abs/1802.03041

[14] B. Biggio and F. Roli, "Wild patterns: Ten years after the rise of adversarial machine learning," *Pattern Recognit.*, vol. 84, pp. 317–331, Dec. 2018.

[15] N. Moustafa, J. Slay, and G. Creech, "Novel geometric area analysis technique for anomaly detection using trapezoidal area estimation on large-scale networks," *IEEE Trans. Big Data*, to be published.

[16] G. Nascimento and M. Correia, "Anomaly-based intrusion detection in software as a service," in *Proc. IEEE/IFIP 41st Int. Conf. Dependable Syst. Netw. Workshops (DSN-W)*, Jun. 2011, pp. 19–24.

[17] J.-H. Lee, M.-W. Park, J.-H. Eom, and T.-M. Chung, "Multi-level intrusion detection system and log management in cloud computing," in *Proc. 13rd Int. Conf. Adv. Commun. Technol. (ICACT)*, 2011, pp. 552–555.

[18] N. Moustafa, B. Turnbull, and K.-K. R. Choo, "An ensemble intrusion detection technique based on proposed statistical flow features for protecting network traffic of Internet of things," *IEEE Internet Things J.*, to be published.

[19] N. Bouguila, D. Ziou, and J. Vaillancourt, "Unsupervised learning of a finite mixture model based on the Dirichlet distribution and its application," *IEEE Trans. Image Process.*, vol. 13, no. 11, pp. 1533–1543, Nov. 2004.

[20] N. Moustafa, G. Creech, and J. Slay, "Big data analytics for intrusion detection system: Statistical decision-making using finite Dirichlet mixture models," in *Data Analytics and Decision Support for Cybersecurity*. Cham, Switzerland: Springer, 2017, pp. 127–156.

[21] (Jul. 2017). *The NSL-KDD Data Set*. [Online]. Available: https://web.archive.org/web/20150205070216/http://nsl.cs.unb.ca/NSL-KDD

[22] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *Proc. IEEE Symp. Comput. Intell. Secur. Defense Appl.*, Jul. 2009, pp. 1–6.

[23] N. Moustafa and J. Slay, "The evaluation of network anomaly detection systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set," *Inf. Secur. J., A Global Perspective*, vol. 25, nos. 1–3, pp. 18–31, 2016.

[24] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Proc. Mil. Commun. Inf. Syst. Conf. (MilCIS)*, 2015, pp. 1–6.

[25] (Jul. 2017). *The UNSW-NB15 Dataset*. [Online]. Available: https://www.unsw.adfa.edu.au/australian-centre-for-cyber-security/cyber%security/ADFA-NB15-Datasets/

[26] L. Huang, A. D. Joseph, B. Nelson, B. I. Rubinstein, and J. Tygar, "Adversarial machine learning," in *Proc. 4th ACM Workshop Secur. Artif. Intell.*, 2011, pp. 43–58.

[27] N. Moustafa and J. Slay. (2017). "A hybrid feature selection for network intrusion detection systems: Central points." [Online]. Available: https://arxiv.org/abs/1707.05505

[28] N. Moustaf and J. Slay, "Creating novel features to anomaly network detection using DARPA-2009 data set," in *Proc. 14th Eur. Conf. Cyber Warfare Secur.*, 2015, p. 204.

[29] M. Barreno, B. Nelson, A. D. Joseph, and J. D. Tygar, "The security of machine learning," *Mach. Learn.*, vol. 81, no. 2, pp. 121–148, 2010.

[30] Z. Tan, A. Jamdagni, X. He, P. Nanda, and R. P. Liu, "A system for denial-of-service attack detection based on multivariate correlation analysis," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 2, pp. 447–456, Feb. 2014.

[31] Z. Tan, A. Jamdagni, X. He, P. Nanda, and R. P. Liu, "Denial-of-service attack detection based on multivariate correlation analysis," in *Proc. Int. Conf. Neural Inf. Process.* Berlin, Germany: Springer, 2011, pp. 756–765.

[32] S. Dua and X. Du, *Data Mining and Machine Learning in Cybersecurity*. Boca Raton, FL, USA: CRC Press, 2016.

[33] U. Amir and K. Hussain, "DDoS attacks detection and prevention techniques in cloud computing: A systematic review," *Int. J. Comput. Sci. Inf. Secur.*, vol. 14, no. 10, p. 390, 2016.

[34] U. Tupakula, V. Varadharajan, and N. Akku, "Intrusion detection techniques for infrastructure as a service cloud," in *Proc. IEEE 9th Int. Conf. Dependable, Autonomic Secure Comput.*, Dec. 2011, pp. 744–751.

[35] S. Subashini and V. Kavitha, "A survey on security issues in service delivery models of cloud computing," *J. Netw. Comput. Appl.*, vol. 34, no. 1, pp. 1–11, 2011.

[36] X. Wang, T.-L. Huang, and X.-Y. Liu, "Research on the intrusion detection mechanism based on cloud computing," in *Proc. Int. Conf. Intell. Comput. Integr. Syst.*, 2010, pp. 125–128.

[37] K. Vieira, A. Schulter, C. Westphall, and C. Westphall, "Intrusion detection for grid and cloud computing," *IT Prof.*, vol. 12, no. 4, pp. 38–43, 2010.

[38] S. N. Dhage and B. Meshram, "Intrusion detection system in cloud computing environment," *Int. J. Cloud Comput.*, vol. 1, nos. 2–3, pp. 261–282, 2012.

[39] A. Zarrabi and A. Zarrabi, "Internet intrusion detection system service in a cloud," *Int. J. Comput. Sci. Issues*, vol. 9, no. 5, pp. 308–315, 2012.

[40] T. Alharkan and P. Martin, "IDSaaS: Intrusion detection system as a service in public clouds," in *Proc. 12th IEEE/ACM Int. Symp. Cluster, Cloud Grid Comput. (CCGRID)*. Washington, DC, USA: IEEE Computer Society, May 2012, pp. 686–687.

[41] P. K. Rajendran, B. Muthukumar, and G. Nagarajan, "Hybrid intrusion detection system for private cloud: A systematic approach," *Procedia Comput. Sci.*, vol. 48, pp. 325–329, Jul. 2015.

[42] J. Nikolai and Y. Wang, "Hypervisor-based cloud intrusion detection system," in *Proc. Int. Conf. Comput., Netw. Commun. (ICNC)*, 2014, pp. 989–993.

[43] Y. Mehmood, M. A. Shibli, A. Kanwal, and R. Masood, "Distributed intrusion detection system using mobile agents in cloud computing environment," in *Proc. Conf. Inf. Assurance Cyber Secur. (CIACS)*, 2015, pp. 1–8.

[44] S. Yi, Z. Hao, Z. Qin, and Q. Li, "Fog computing: Platform and applications," in *Proc. 3rd IEEE Workshop Hot Topics Web Syst. Technol. (HotWeb)*, Nov. 2015, pp. 73–78.

[45] Y. Shi, S. Abhilash, and K. Hwang, "Cloudlet mesh for securing mobile clouds from intrusions and network attacks," in *Proc. 3rd IEEE Int. Conf. Mobile Cloud Comput., Services, Eng.*, Mar./Apr. 2015, pp. 109–118.

[46] Q. Yaseen, F. Albalas, Y. Jararwah, and M. Al-Ayyoub, "Leveraging fog computing and software defined systems for selective forwarding attacks detection in mobile wireless sensor networks," *Trans. Emerg. Telecommun. Technol.*, vol. 29, no. 4, p. e3183, 2018.

[47] R. Sandhu, A. S. Sohal, and S. K. Sood, "Identification of malicious edge devices in fog computing environments," *Inf. Secur. J., A Global Perspect.*, vol. 26, no. 5, pp. 213–228, 2017.

[48] S. Mei and X. Zhu, "Using machine teaching to identify optimal training-set attacks on machine learners," in *Proc. AAAI*, 2015, pp. 2871–2877.

[49] H. Xiao, B. Biggio, G. Brown, G. Fumera, C. Eckert, and F. Roli, "Is feature selection secure against training data poisoning?" in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1689–1698.

[50] P. J. Rousseeuw and M. Hubert, "Robust statistics for outlier detection," *Data Mining Knowl. Discovery*, vol. 1, no. 1, pp. 73–79, 2011.

[51] L. Harasim, *Learning Theory and Online Technologies*. New York, NY, USA: Taylor & Francis, 2017.

[52] W. Fan, N. Bouguila, and D. Ziou, "Variational learning for finite Dirichlet mixture models and applications," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 5, pp. 762–774, May 2012.
[53] W. Fan, N. Bouguila, and Ziou, "Unsupervised anomaly intrusion detection via localized Bayesian feature selection," in *Proc. IEEE 11th Int. Conf. Data Mining (ICDM)*, Dec. 2011, pp. 1032–1037.
[54] M. D. Escobar and M. West, "Bayesian density estimation and inference using mixtures," *J. Amer. Stat. Assoc.*, vol. 90, no. 430, pp. 577–588, 1995.
[55] S. Gensler, "Finite mixture models," in *Handbook of Market Research*. Cham, Switzerland: Springer, 2017, pp. 1–14.
[56] B. S. Oboh and N. Bouguila, "Unsupervised learning of finite mixtures using scaled Dirichlet distribution and its application to software modules categorization," in *Proc. IEEE Int. Conf. Ind. Technol. (ICIT)*, Mar. 2017, pp. 1085–1090.
[57] R. Vidal, Y. Ma, and S. S. Sastry, "Principal component analysis," in *Generalized Principal Component Analysis*. New York, NY, USA: Springer-Verlag, 2016, pp. 63–122.
[58] X. Wan, W. Wang, J. Liu, and T. Tong, "Estimating the sample mean and standard deviation from the sample size, median, range and/or interquartile range," *BMC Med. Res. Methodol.*, vol. 14, no. 1, p. 135, 2014.
[59] (Sep. 2018). *MySQL Cluster CGE*. [Online]. Available: https://www.mysql.com/products/cluster/
[60] T. Duong, "KS: Kernel density estimation and kernel discriminant analysis for multivariate data in R," *J. Stat. Softw.*, vol. 21, no. 7, pp. 1–16, 2007.
[61] W. Liu, P. P. Pokharel, and J. C. Principe, "Correntropy: Properties and applications in non-Gaussian signal processing," *IEEE Trans. Signal Process.*, vol. 55, no. 11, pp. 5286–5298, Nov. 2007.
[62] Z. Tan, A. Jamdagni, X. He, P. Nanda, R. P. Liu, and J. Hu, "Detection of denial-of-service attacks based on computer vision techniques," *IEEE Trans. Comput.*, vol. 64, no. 9, pp. 2519–2533, Sep. 2015.
[63] C.-F. Tsai and C.-Y. Lin, "A triangle area based nearest neighbors approach to intrusion detection," *Pattern Recognit.*, vol. 43, no. 1, pp. 222–229, 2010.
[64] P. Saurabh and B. Verma, "An efficient proactive artificial immune system based anomaly detection and prevention system," *Expert Syst. Appl.*, vol. 60, pp. 311–320, Oct. 2016.
[65] M. A. Ambusaidi, X. He, P. Nanda, and Z. Tan, "Building an intrusion detection system using a filter-based feature selection algorithm," *IEEE Trans. Comput.*, vol. 65, no. 10, pp. 2986–2998, Oct. 2016.

**Nour Moustafa** received the bachelor's and master's degrees in computer science from Helwan University, Egypt, in 2009 and 2014, respectively, and the Ph.D. degree in cyber security from the University of New South Wales (UNSW), Canberra, in 2017. He is currently a Lecturer with UNSW and Helwan University. His research interests include cyber security, in particular, network security, host- and network-intrusion detection systems, statistics, deep learning, and machine learning techniques. He is interested in designing and developing threat detection and forensic mechanisms to the Industry 4.0 technology for identifying malicious activities from cloud computing, fog computing, IoT, and industrial control systems over virtual machines and physical systems.

**Kim-Kwang Raymond Choo** (SM'15) received the Ph.D. degree in information security from the Queensland University of Technology. He currently holds the Cloud Technology Endowed Professorship with The University of Texas at San Antonio, and has a courtesy appointment at the University of South Australia. He is also a Fellow of the Australian Computer Society. In 2015, he and his team won the Digital Forensics Research Challenge organized by the University of Erlangen-Nuremberg, Germany. In 2016, he was named the Cybersecurity Educator of the Year-APAC (Cybersecurity Excellence Awards are produced in cooperation with the Information Security Community on LinkedIn). He was a recipient of the British Computer Society's Wilkes Award in 2008, Australia Day Achievement Medallion, Fulbright Scholarship in 2008 and 2009, the 2014 Highly Commended Award by the Australia New Zealand Policing Advisory Agency, the ESORICS 2015 Best Research Paper Award, the 2018 UTSA College of Business Col. Jean Piccione and Lt. Col. Philip Piccione Endowed Research Award for Tenured Faculty, and the IEEE TrustCom 2018 Best Paper Award. He is a Co-Chair of the IEEE Multimedia Communications Technical Committee's Digital Rights Management for Multimedia Interest Group, and an Honorary Commander of the 502nd Air Base Wing and Joint Base San Antonio-Fort Sam Houston.

**Ibrahim Radwan** received the Ph.D. degree in computer science from the University of Canberra in 2015. From 2014 to 2016, he was a researcher in a leading automotive industry warehouse. He is currently a Research Fellow with The Australian National University. His research includes developing and implementing algorithms in computer vision, machine learning, robotics, and artificial intelligence.

**Seyit Camtepe** (SM'17) is a senior research scientist at CSIRO Data61. He received the Ph.D. degree in computer science from Rensselaer Polytechnic Institute, New York, NY, USA, in 2007. He had six years of industry experience as a Network and Security Engineer, prior to receiving the Ph.D. degree. From 2007 to 2013, he was a Senior Researcher and a Research Group Leader in security with the Technische Universitaet Berlin, Berlin, Germany. Since 2013, he has been with Queensland University of Technology (QUT), Brisbane, QLD, Australia, as a Lecturer (Assistant Professor) with the School of Electrical Engineering and Computer Science. His current research interests include pervasive and autonomous security, applied and malicious cryptography, and IoT security.