IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 24, 2019, accepted December 6, 2019, date of publication December 25, 2019, date of current version January 6, 2020.

Digital Object Identifier 10.1109/ACCESS.2019.2961609

Efficient Non-Linear Covert Channel Detection in TCP Data Streams

HANAA NAFEA[1,2], KASHIF KIFAYAT[3], QI SHI[1], KASHIF NASEER QURESHI[4], AND BOB ASKWITH[1]

[1]Department of Computer Science, Liverpool John Moores University, Liverpool L3 3AF, U.K.
[2]Department of Computer Science, College of Computer Science and Engineering, Taibah University, Medina 4144, Saudi Arabia
[3]Department of Cyber Security, Air University, Islamabad 44000, Pakistan
[4]Department of Computer Science, Bahria University, Islamabad 44000, Pakistan

Corresponding author: Hanaa Nafea (h.o.nafea@2014.ljmu.ac.uk)

This work was supported by Taibah University.

ABSTRACT Cyber-attacks are causing losses amounted to billions of dollars every year due to data breaches and vulnerabilities. The existing tools for data leakage prevention and detection are often bypassed by using various different types of sophisticated techniques such as network steganography for stealing the data. This is due to several weaknesses which can be exploited by a threat actor in existing detection systems. The weaknesses are high time and memory training complexities as well as large training datasets. These challenges become worse when the amount of generated data increases in every second in many realms. In addition, the number of false positives is high which makes them inaccurate. Finally, there is a lack of a framework catering for the needs such as raising alerts as well as data monitoring and updating/adapting of a threshold value used for checking the data packets for covert data. In order to overcome these weaknesses, this paper proposes a novel framework that includes elements such as continuous data monitoring, threshold maintenance, and alert notification. This paper also proposes a model based on statistical measures to detect covert data leakages, especially for non-linear chaotic data. The main advantage of the proposed model is its capability to provide results with tolerance/threshold values much more efficiently. Our experiments indicate that the proposed framework has low false positives and outperforms various existing techniques in terms of accuracy and efficiency.

INDEX TERMS Data leakage, network steganography, covert channel, TCP/IP protocol.

I. INTRODUCTION

The rapid expansion of Information and Communication Technologies (ICTs) and highly interconnected critical infrastructures have created various cyber security threats and challenges. The majority of global trade is done through the Internet for fast and cost effective services. Security is one of the significant factors to protect these online businesses and services from different kinds of cyber-attacks. Security attacks are causing losses of billions of dollars every year and getting more sophisticated day by day. Data leakage can be defined as the intentional or unintentional disclosure of confidential or private data to a party who does not have permission to access it [1]. Steganography is a covert communication technique. In network steganography, different communication protocols such as Transmission Control Protocol (TCP) [2], Internet Protocol (IP) [3], Hypertext Transfer Protocol (HTTP) [4], Real-time Transport Protocol (RTP) and RTP Control Protocol (RTCP) [2] are used for covert communication. A covert channel is a communication channel that is neither designed nor intended to exist, and it can be used to transfer information in a manner that violates a given security policy.

Based on existing literature, covert channel detection techniques are categorised into approaches based on pattern, machine learning and statistics. The pattern-based approaches [5] have the common denominator of matching a captured packet header against a known pattern in order to identify if the packet has covert data in the header. These techniques provide the benefit of high accuracy and the speed of detection, but they suffer from the inability to cope with new threat vectors and incomplete datasets, resulting in high maintenance requirements. The machine learning based approaches [6] use a set of rules created from

The associate editor coordinating the review of this manuscript and approving it for publication was Tawfik Al-Hadhrami.

the classification of complex datasets. These overcome some shortcomings of the pattern based approaches by improving the ability to cope with incomplete datasets and offering insights into latent information within the datasets. However, the benefits are offset by the shortcomings of large system footprints, low detection speed and high maintenance requirements. Finally, the statistical approaches [7] measure deviations from expected stochastic behaviour, which effectively means that they are also capable of handling incomplete datasets. Additionally, they offer high detection speed and ability to detect new threat vectors. The main shortcoming of these approaches, however, is the possibility of false positive results.

It has been identified that the common problem with existing approaches is the inability to adapt in order to detect new attack vectors, particularly when the attack vectors are masked, i.e., the data leakage occurs via the fields in packet headers where the data can be non-linear [3], [8]. A non-linear type of covert data means that the change of output is not proportional to the change of input. Unlike linear covert data with rather simple dynamics, non-linear covert data has very complex dynamical behavior, so it is not easy to reverse-engineer the covert data to its original form. When non-linear data is sensitive to initial conditions, it becomes non-linear chaotic data. Sensitivity to the initial conditions means that each point in a chaotic system is arbitrarily and closely approximated by other points. A classic example is that of double pendulums where a slight difference between the points of the release of the two pendulums causes a large difference in the trajectories taken by them. Effectively, this means that machine learning or pattern finding based approaches may not be suitable as they work in the realms of fixed pattern locations. Thus the pattern and machine learning based approaches fail in such cases, as there is no fixed pattern and the classification of non-linear data is mathematically not viable [3], [9]. Statistical approaches do offer some degree of effectiveness, but again similar to machine learning based approaches, they fail due to the chaotic nature of non-linear data being leaked via packet headers [10]. Overall, this makes it difficult for existing techniques to differentiate between normal data and the data masked to appear as normal data. Additionally, it is also observed that various techniques suggested in the literature fail to cope with the diversity of fields in the TCP header and it is this inability that makes them unsuitable for coping with new threats [9].

In this paper we therefore propose:

- A covert channel detection framework for TCP, which can be used to detect concealed data in a multitude of fields in a TCP header in real-time, whilst having a very small system footprint. The proposed framework is novel in that not all deviation scores from a decision maker are tagged as data leakage incidents; rather, our proposed solution focuses on further analysis using a novel approach to the quantification of skewed results, which informs the probability of data leakage. The process uses the means of moving average for making decisions about data leakage, thereby increasing the level of accuracy as well as reducing the likelihood of Type I (asserting that something is true when it is actually false) and Type II (asserting that something is false when it is actually true) errors. Moving average offers the smoothening of the erratic data and provides the ability to observe trends that help reduce the likelihood of the aforementioned errors.
- The use of a statistical threshold calculation technique, which adopts a hybrid (based on regressive auto-correlation and classical statistics) approach to calculation that improves on the limitations observed in existing covert data leakage detection techniques. The resultant calculated threshold value is based on expected values across multiple operating systems, thus ensuring longevity with regard to its validity and thereby ensuring that various flavours of operating systems are not susceptible to leakage.
- A statistical technique that quantifies the level of covert communication by computing the deviation of observed data from an expected threshold value. The technique is a two-stage analysis process where a deviation score is calculated in real time over a buffer of collected TCP headers as well as over a larger collection of TCP headers. This offers insights into temporal data leakage detection for larger data collections, informing the analyst of time based indicative analysis as well as on-the-fly outlier analysis, which are both improvements on existing detection techniques. Note that although existing statistical techniques offer probabilistic matching, they fail to accommodate non-linear data of chaotic nature. A statistical threshold calculation technique, which adopts a hybrid (based on regressive auto-correlation and classical statistics) approach to calculation, improves on the limitations observed in the existing techniques.

The paper is organized as follows. In Section II we provide the background information about TCP/IP based steganography and a brief description of related works within the context of covert channel detection. Section III presents our proposed covert data monitoring framework. Section IV specifies our statistical algorithm for folding non-linear chaotic random data. Section V provides a description about the maintenance of thresholds - an algorithm for the creation of thresholds. Section VI presents an algorithm used to quantify outliers (i.e., covert data). Section VII describes our experiments, the evaluation of the experiment results and a comparison of our work to existing approaches in the field of covert data detection. Finally, our conclusion is given in Section VIII.

II. BACKGROUND AND RELATED WORK

Steganography can be implemented in different fields in TCP/IP. However, in the past, stenographic techniques have failed in TCP because of a different probability distribution to that of the unmodified TCP/IP implementations. According to [11], the TCP/IP header is highly susceptible to steganography. This comprises header fields such as Type of service,

IP flags, Fragments offset, IP options, TCP timestamp, Packet order, Packet timing and TCP initial sequence number. For details please refer to [11] that provides descriptions per field. The existing research has shown that covert channels are divided into three major categories, i.e., covert timing channels, covert storage channels and covert network channels [12]. A covert timing channel takes advantage of the performance of a system component to send a secret message. Storage based covert channels use the method of sending intended hidden information by using the common area between processes in memory, or say, the main memory of the system. The main aim of covert network channels is to hide secret data inside the carriers/packets, i.e., normal network traffic of users. In an ideal situation, hidden data exchange cannot be detected by third parties unaware of the covert channel usage [2]. The HTTP and TCP/IP protocols are linked to known types of covert network channels. For example, in the TCP/IP type of covert network channel, TCP/IP based modification performs stenographic covert channels over network protocol header values to conduct secret communication [9], [12], [13].

In order to prevent data leakage, authors in [14] have formulated data leakage prevention problems as Partially Observable Markov Decision Processes. The model created encodes a mechanism for monitoring that is not visible to the recipients of the data. The technique has been called digital watermarking. The research goal of [14] is to create optimal information sharing strategies for the sender and optimal information leakage strategies for a malicious recipient as a function of the ability of the monitoring mechanism. The experiments show the effects of different settings on the cost of data leakage, fuzziness level and trust of the sender in recipients.

Statistical techniques involve data capturing followed by the measurement of deviation from expected stochastic behaviour, i.e., they are appropriate where observations are not reproducible exactly. Other measurements include the audit of records, categorisation of various activities, amount of activity and also system resource usage. The deviations are calculated by comparing the resultant to preknown/calculated stochastic expectation. If the measures are outside the postulated thresholds, the result is categorised as failure/leakage activity. There are other approaches where failure is linked to the level of irregularity in the data under observation and it is considered as an outlier if it is not within postulated thresholds. Various statistical techniques can be categorised into the approaches of such Univariate Analysis Models and Multivariate Analysis Models. In a Univariate Analysis Model analysis, the measurements are obtained from an individual variable or attribute to measure the overall irregularity. In Multivariate Analysis Model analysis, the measurements are obtained from more than one variable. Techniques such as Principal Component Analysis, Multiple Regression Analysis or Partial Least Squares are used to identify the prevailing patterns in the data that are categorised as trends and hence depict outliers that are being monitored. This type of analysis has the known advantage of reducing the probability of the Type I error and decomposing correlated measurements into a new set of uncorrelated measurements that is beneficial for pattern recognition.

Various detection techniques for covert channels have various costs in terms of the processing power and sensitivity of execution. Protocol based detections are simple to implement and have low processing power requirements. However, they can only detect badly written implementations. Signature based detection is more sensitive and raises a notification when protocol implementation signatures do not match. Processing requirements for signature-based detection are higher than those for protocol based detection. However, they are not as high as the requirements for the behaviour-based detection of malicious packets. Signature-based detection is the most sensitive and efficient technique, but it has a very high false positive rate.

III. PROPOSED COVERT DATA MONITORING FRAMEWORK

In the previous section, we discussed the extent of the shortcomings of existing solutions and indicated the challenges in creating a solution for covert data monitoring on TCP headers within the realms of network communication. In order to resolve the above, we have devised a novel Covert Data Monitoring Framework to specifically monitor complex, nonlinear, chaotic, random and dynamic data with a degree of uncertainty. It focuses mainly on protection against data leakage over covert channels. The framework also focuses on the detection of both deliberate and accidental covert data leakage and endeavours to notify the relevant authorised parties about detected incidents.

The framework is a hybrid one, bringing together various components that operate and reside on the hosts protected against data leakage. There is a degree of collaboration between various host components in order to ensure that the framework's threshold profiler remains updated autonomously. Furthermore, it has been designed to be self-contained with a small system footprint, while ensuring that its detection speed remains high and the results remain effective. The framework uses a statistical approach for monitoring, detection and decision-making. The framework resides on the top of an operating system and has the capability to obtain data from the network layer and to utilize operating system functions such as notifications, as illustrated in Figure 1. The proposed framework operates by monitoring incoming network data in real time and comparing it with the threshold profiles managed by the threshold profiler component. The threshold profiles are designed specifically to cope with dynamic, non-linear, chaotic and random data and they are calculated using a novel threshold calculation/adaptation algorithm. It is by comparing the real time data against the threshold values so that a small modal change in the header field values can indicate a leakage, which also takes a small amount of time. The framework has been designed to refine the threshold profiles periodically, using the threshold

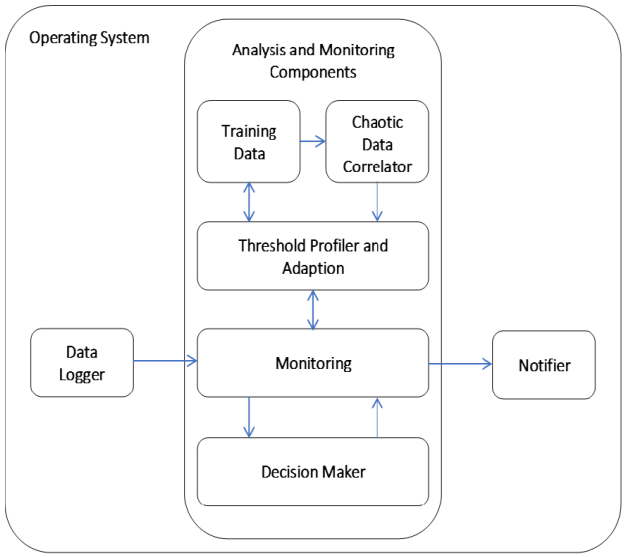


FIGURE 1. High level architecture of covert data monitoring framework.

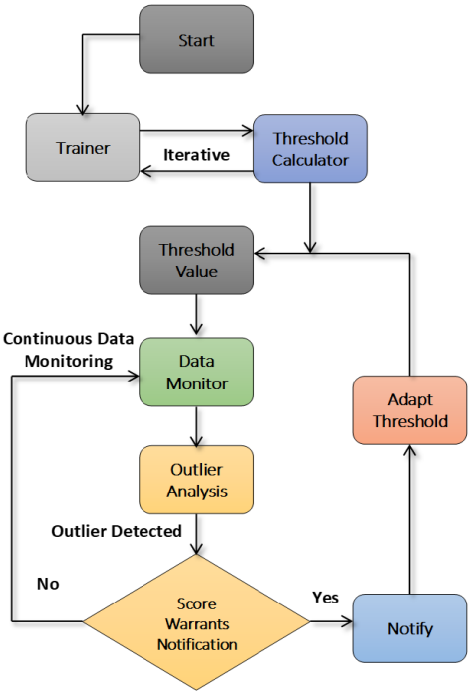


FIGURE 2. Covert data monitoring framework at runtime.

adaptation algorithm. This adaptation allows the framework to evolve and also ensures that unknown data leakage strategies and attacks can be combated. The framework is so distinct in that not all deviation scores derived by the decision maker component are flagged as data leakage incidents; these scores are further analysed using a novel approach to the quantification of skewed results, which informs the probability of data leakage.

The various operations of the framework have been categorized into six different phases illustrated using different colours in Figure 2. Note that the flowchart attempts to present a successful time scenario and potential consequences without depicting the cases of failure at runtime. These phases are described in detail below.

Trainer (Light Grey): It is part of the training phase. During the initial training stage, the framework will initiate the computation of an optimal number of dimensions in order to find the correlation between the various points of given data series. Note that this phase is only triggered once for a data type in the ISN (Initial Sequence Number) field and no subsequent computation is required unless a data type changes.

Threshold Calculator (Blue): It is also part of the training phase. During the threshold calculation stage, the system collects the network data and stores them in a persistent storage. Note that it is assumed that the data collected at this stage is free of any covert data leakage activity. In order to calculate the threshold, an iterative process is required that would consume trainer data and compute a threshold value.

Threshold Value (Dark Grey): The threshold value is stored in this phase in a persistent storage. Note that the threshold value can be computed by the Threshold Calculator using the trainer data or adapted via the Adapt Threshold phase as a result of outlier detection.

Data Monitor (Green): This phase of the framework is where the real time monitoring of TCP packets is conducted. The monitoring includes the comparison of a computed third order feature variance value to the threshold value from the previous phase. Note that during runtime this phase does not require a large number of hardware resources as the comparison is conducted with a small buffered set of incoming TCP packets.

Outlier Analysis (Yellow): Once the data monitoring phase raises an alarm regarding a breach of the threshold value, the statistical outlier analysis phase is started. This phase uses deviation to calculate the amount of movement in the third order feature variance value and computes a moving average value by comparing it to historic values computed over the last four periods. The notification phase will begin only if the average value is skewed towards a breach; otherwise, the alarm will be recorded but no notification event will be raised. This is to reduce the amount of Type I and Type II errors.

Notify (Light Blue): This notification phase proactively raises alarms and also records a data leakage breach. It is one of the requirements of the framework. Notification methods include calls to low level functions of the operating system to take remedial actions, to disable network interfaces and functions if necessary, and to send alert emails to the administrator of the system being monitored.

Adapt Threshold (Orange): The threshold adaptation phase includes threshold recalculation if the amount of skewness found during the outlier analysis is larger than expected. The adaptation forms a closed loop whereby the threshold value can be recomputed proactively to increase the likelihood of detecting outliers and to improve the overall accuracy of the framework. The newly adapted threshold value is stored in the Threshold Value persistent storage.

IV. PROPOSED STATISTICAL ALGORITHM FOR COVERT CHANNEL DETECTION

As described in [15] and in Section II, the ISNs generated by operating systems are not truly random; they are pseudo random numbers that have initial relation to ISNs generated in the past. In order to forecast a future value in series, there are a number of forecasting methods, e.g. trend curve based prediction for linear systems as well as logarithmic regression, and Auto-Regression Integrating Moving Averages (ARIMA) [15] for non-linear data. ARIMA is used to compute the number of initial values (dimensions) that are used to computer a future value. In our case, ARIMA is employed to compute the embedding dimension that is the number of past values used to predict future values so that we can calculate the next ISN value to compare it to the ISN value within the packet. The comparison is critical, because the packet would be considered as an outlier if the values are different beyond the threshold value.

A. MAINTENANCE OF THRESHOLDS

This section discusses the components required for the statistical algorithm used for marinating thresholds. Figure 3 illustrates its process. The sequence number in the TCP header comprises a random initial number used for the communication between two network nodes. It is mandatory and has a maximum value of 2^{32}.

In order to ensure the integrity of TCP/IP connection, every stream is assigned a unique random sequence number. This is done so that attackers may not be able to perform blind spoofing (a practice where an ISN can be guessed) and change the data integrity of a packet. According to [16], it is difficult to generate an unpredictable number using a computer. The reason for this is that computers are designed to strictly execute a defined set of commands in a repeatable and accurate way. A fixed algorithm is used to produce exactly the same result on a different computer that can hence predict output values (provided that the internal state of a remote system is accurately reconstructed) [16]. It is observed through a pseudo-random number generator (PRNG) that the algorithm will start generating the same set of sequences over again because of a limited number of internal states that can be used by the algorithm [15]. It is observed that PRNG used for sequence numbers in TCP headers follows patterns (based on their different implementations) in operating systems [13]. This has been spotted through Phase Space Analysis [16]. It was observed that a correlation between subsequent results is followed when generating random numbers. The aforementioned property between generated random numbers can be used to find out if a subsequent random number is correlated in a specific way. For the purpose of creating TCP/IP packets, a PRNG is used to generate a sequence of numbers that approximate to properties of random numbers. It is the randomness of ISNs that makes it difficult for attackers to predict them. However, the fact that random numbers generated using an algorithm are actually pseudo random makes the communication open to vulnerability. It is the uniqueness

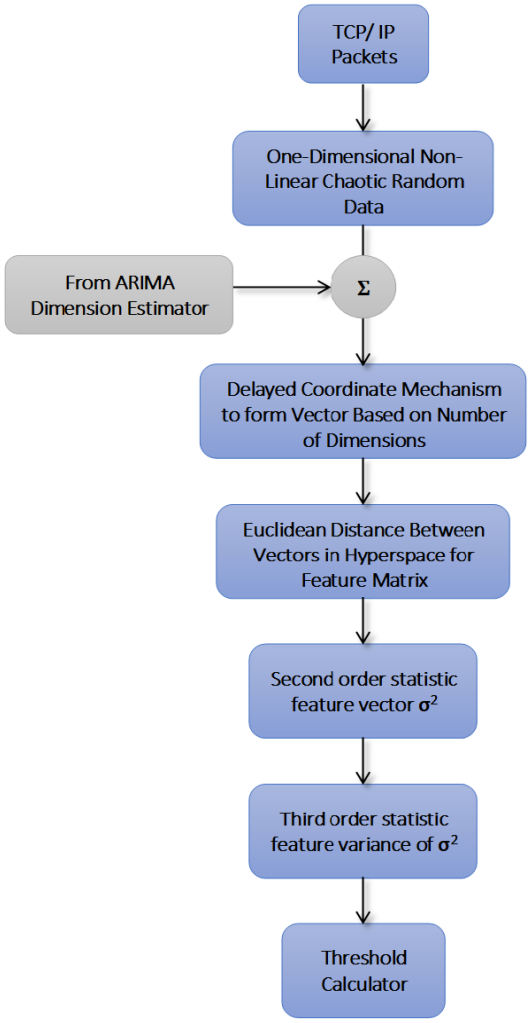


FIGURE 3. Flowchart for the creation of features and the threshold calculation for a profile - Training Process. The figure shows the process followed for the maintenance of thresholds, where the one dimesional non-linear data is expanded in terms of previous values to compute the threshold value.

of ISNs within a given time frame, which ensures that the fragments of different packets are not assembled into one packet at the receiving end. A PRNG in operating systems is modelled as a function of which the input is a short random seed and the output is indistinguishable from truly random bits. Various implements of a PRNG exist that implement a deterministic function [17]. Sheela and Sathyanarayana [18] reported that pseudo random number generators are derived from a deterministic, chaotic and dynamic system, thus making a connection between chaos and pseudo random number generators.

Phase space reconstruction is a very useful non-linear or chaotic signal processing technique to find out about a dynamic system. Topologically, the phase space and original system are equivalent and hence it is possible to recover the non-linear dynamics of a generating system [19]. The implication is that the complete dynamics of the system are accessible in this space.

V. PROPOSED STATISTICAL THRESHOLD CREATION ALGORITHM

The starting point for the analysis of any non-linear dataset (here, for instance, initial sequence numbers generated for packet creation in TCP/IP communication) is the construction of a phase space or the creation of a portrait of the phase space. The state of the system is described as the state of its variables, and n state variables observed at time t form a vector in an n dimensional space called a phase space. The state of the system typically changes with time and hence the vector in the phase space describes the trajectory of the system or the evolution/dynamics of the system. It is this shape of trajectory that has hints/indications about the system. Chaotic or periodic systems have characteristics in the phase space. To reveal the hidden structure of a random number phase space, its construction using "delay coordinates" is widely used. For given *ISN(i)* numbers, the phase space (datasets) is constructed as follow:

$$Y_i = (ISN(i), ISN(i-1), \ldots, ISN(i-(m-1)) \quad (1)$$

Here, $i = 1, 2, \ldots .i - m + 1$, where i is the number of ISNs, and m is the dimension. Vector Y_i is the new phase space (dataset) that is formed from time-delayed values of the initial ISN value scalar measurements. The first order difference (as shown in Equation 2) is constructed as follows [15]:

$$\begin{aligned} \chi(n) &= ISN(n) - ISN(n-1) \\ y(n) &= ISN(n-1) - ISN(n-2) \\ z(n) &= ISN(n-2) - ISN(n-3) \\ w(n) &= ISN(n-3) - ISN(n-4) \end{aligned} \quad (2)$$

Here, $n = N, N-1, N-2, \ldots, 5$, and $x(n), y(n), z(n)$ and w(n) are point coordinates used to create a phase space dataset.

Chaos theory dictates that phase space vectors are fully representative of the non-linear dynamics of an original dataset, when the embedding dimension m is large enough.

For an optimal embedding m, according to [19], the observation of a real process generally does not yield all of its state variables. This is generally either because not all state variables are known or because not all of them can be measured.

It is clear that the numbers generated are clearly the result of unknown regression with dependent components. As they are unknown, it can be assumed that they are formed of independent variables $y_1, y_2, y_3 .., y_t$ where y_{t+1} is dependent upon the value of y_t. This type of time series is known as a univariate time series (a series with single observations recorded over regular intervals). In a univariate time series, the past value of an independent variable is used to calculate the value of a new independent variable as depicted in the following equation:

$$y_t = \beta y_{t-1} + \varepsilon \quad (3)$$

Here, β is a constant and ε is an error value. The above equation explains that sometimes when the set of explanatory variables required by a regression model is unavailable (true for random numbers generated as a function of time with unknown other variates), then it becomes a beneficial choice to use only a single variable to forecast future values.

Among the models available for modelling a univariate series are:

- Auto-Regressive Model (AR)
- Moving Average Model (MA)
- Auto-Regressive Moving Average Model (ARMA)
- Auto-Regressive Integrating Moving Average Model (ARIMA)

In the AR model, y_t depends only on its own past values y_{t-1}, y_{t-2}, y_{t-3}, etc. Thus we have $y_t = f(y_{t-1}, y_{t-2}, y_{t-3}, \ldots, \varepsilon_t)$, where ε_t is the noise or error term. A common representation of an auto-regressive model depending on p past values is called an *AR(p)* model and represented as:

$$y_t = \beta_0 + \beta_1 y_{t-1} + \beta_2 y_{t-2} + \cdots + \beta_p y_{t-p} + \varepsilon_t \quad (4)$$

For Equation 4, it is important to know that the value of p is about how far back in time the value of y should be picked in order to estimate y_t. Generally, in a real life phenomenon, it has been observed that past values up to 3 steps (i.e., forming an *AR(3)* model) are sufficient [20]. In this context, it is critical to compute p so that accuracy can be obtained.

In the MA model, y_t depends only on its error terms ε_{t-1}, $\varepsilon_{t-2}, \varepsilon_{t-3}$, etc. A common representation of a moving average model where it depends on q past values is called a *MA(q)* model and defined as:

$$y_t = \beta_0 + \varphi_1 \varepsilon_{t-1} + \varphi_2 \varepsilon_{t-2} + \cdots + \varphi_q \varepsilon_{t-q} + \varepsilon_t \quad (5)$$

The ARMA model refers to a combined usage of the AR and MA models, denoted as ARMA(p, q). Hence, it is represented as:

$$\begin{aligned} y_t = \beta_0 + \beta_1 y_{t-1} + \beta_2 y_{t-2} + \cdots + \beta_p y_{t-p} \\ + \varphi_1 \varepsilon_{t-1} + \varphi_2 \varepsilon_{t-2} + \cdots + \varphi_q \varepsilon_{t-q} + \varepsilon_t \end{aligned} \quad (6)$$

Auto-correlation (ACF) refers to the way in which the observations in a series are related to each other and measured by simple correlation between the current observation (y_t) and the observation in the p^{th} period from the current one (i.e., y_{t-p}). It is defined as:

$$\rho_k = Corr\left(y_t, y_{t-p}\right) = \frac{Cov\left(y_t, y_{t-p}\right)}{\sqrt{var\left(y_t\right)}\sqrt{var\left(y_{t-p}\right)}}$$

Here, *Cov* is covariance and *var* is variance. p_k informs about how many periods back one should look into for creating stationary series.

Partial Correlation (PACF) refers to the degree of correlation between y_t and y_{t-p}, which is defined as:

$$\begin{aligned} \rho_k &= Corr\left(y_t, y_{t-p}\right) \\ &= \frac{Cov\left(y_t, y_{t-p} | y_{t-p-1}, y_{y-p-2}\right)}{\sqrt{var\left(y_t | y_{t-p-1}, y_{t-p-2}\right)}\sqrt{var\left(y_{t-p} | y_{t-p-1}, y_{t-p-2}\right)}} \end{aligned}$$

The ACF or PACF is available for various values of lags of auto-regressive and moving average components,

i.e., p and q. Based on Equation 2 for the four dimensional phase vector, r_i is constructed as:

$$r_i = [x(i), y(i), z(i), w(i)], \quad i = 1, 2, \ldots, M, \; M = N - 4 \tag{7}$$

Let R represent the phase space dataset formed from Equation 2 and Equation 7. The number of phase space vectors in R is N−4.

$$R = [r_1, r_2, \ldots, r_M] \tag{8}$$

In order to extract features / patterns from R, the distance between two vectors r_i and r_j is calculated in the phase space as follows:

$$d_{i,j} = \sqrt{(x(i)-x(j))^2+(y(i)-y(j))^2+(z(i)-z(j))^2+(w(i)-w(j))^2} \tag{9}$$

Computing the distances between any two vectors in R forms a 2-dimensional matrix D as shown below:

$$D = \begin{pmatrix} d_{1,1} & d_{1,2} & \cdots & d_{1,M} \\ d_{2,1} & d_{2,2} & \cdots & d_{2,M} \\ \vdots & \vdots & \cdots & \vdots \\ d_{k,1} & d_{k,2} & \ldots & d_{k,M} \\ \vdots & \vdots & \cdots & \vdots \\ d_{M,1} & d_{M,2} & \cdots & d_{M,M} \end{pmatrix} \tag{10}$$

Equation 10 shows a 2 dimensional distance matrix formed by the Euclidean distances between the two vectors in R, with $k = 1, 2, \ldots, M$. A row vector represents the distances between a specified vector and all the vectors in R, hence the row size is M. As the distances between the vector and itself will always be zero, it makes the diagonal entries of D zeros.

$$d(k) = \left[d_{k,1}, d_{k,2}, \ldots, d_{k,M}\right], \quad j = 1, 2, \ldots, M \tag{11}$$

Note that in Equation 11, $d_{k,k} = 0$. The variance of row vector d_k is calculated as follows:

$$\sigma_k^2 = \frac{1}{M-1} \sum_{i=1}^{M} \left(d_{k,i} - \mu_k\right)^2, \quad i, k = 1, 2, \ldots, M \tag{12}$$

Here, μ_k is the mean value of row vector $d(k)$. σ_k^2 represents the variance of the Euclidean distances between the specified vector r_k and all other vectors in R. Upon calculation of the variance, we obtain the variance vector σ^2 as shown in Equation 13.

$$\sigma^2 = \left[\sigma_1^2, \sigma_2^2, \ldots, \sigma_M^2\right] \tag{13}$$

Furthermore, Equation 14 below calculates the variance of vector σ^2. The computed variance is based on 1000 ISNs that can be obtained by an ISN training sequence. Here μ_σ represents the mean value of those in σ^2. It is expected that the variation in distance will be of the same order, so 10% of the variance has been selected as the threshold as shown in Equation 15. The justification of 10% as the threshold is to ensure that the false positive rate remains low.

$$var_\sigma = \frac{1}{M-1} \sum_{i=1}^{M} \left(\sigma_i^2 - \mu_\sigma\right)^2, \quad i = 1, 2, \ldots, M \tag{14}$$

$$Threshold_{ISN} = \frac{var_\sigma}{10} \tag{15}$$

VI. PROPOSED STATISTICAL ALGORITHM FOR QUANTIFICATION OF OUTLIERS

With a header field that can take random data, it becomes highly dynamic and difficult to monitor for covert data leakage. Therefore, when a threshold breach is reported, it does not automatically indicate a data leakage event. Instead, every event (in which a deviation from the threshold is noted) is subjected to further analysis to quantify the outlier. Our proposed quantification algorithm provides a score to a detected outlier, which forms a dual stage analysis process. In the first stage, the actual event of outlier detection is reported. In the second stage, the extent of the breach is measured by scoring the deviation in relation to the threshold. This improves the accuracy of the detection and reduces the likelihood of Type I and Type II errors. Figure 4 shows the flowchart detailing the process that quantifies and decides if a certain packet is legal or illegal.

VII. EXPERIMENTS AND RESULTS

A. DATASET AND SETUP OF STATIONARY SERIES

Within a TCP packet header, there are a number of fields that can be used for potential covert data leakage. One of the fields, which is highly dynamic and therefore tends to be a favoured choice for data leakage due to its inherent need for random values, is the ISN field. As discussed in Section IV, an ISN is a random number generated using a PRNG. However, such random numbers are produced through a sequence of numbers, so in reality they are not truly random numbers. Attackers can predict these numbers. A PRNG is normally used in Window, Linux and other Operating Systems. Due to the coupling that exists between the system's components, we can reconstruct the system's phase space trajectory from a single observation by using a time delay mechanism (delayed coordinate) as suggested by Takens et al. in [19]. This reconstruction of the phase space is called time delay embedding. Figure 5 and Figure 6 show the ACF and PACF plots for a stationary ISN dataset at $m = 4$. It is noticed that ACF and PACF reside outside thresholds, indicating that the auto-regression of some order will be required. This has also been tested using the Dickey-Fuller Test (See Figure 7) to prove that the series has been made stationary, which extracts a larger set of temporal patterns useful for our analysis.

B. TEST AND RESULTS

The setup of our experiments was conducted on Ubuntu Linux VM that was mounted using Oracle Virtual Box on a Windows 7 computer. The choice of the operating system

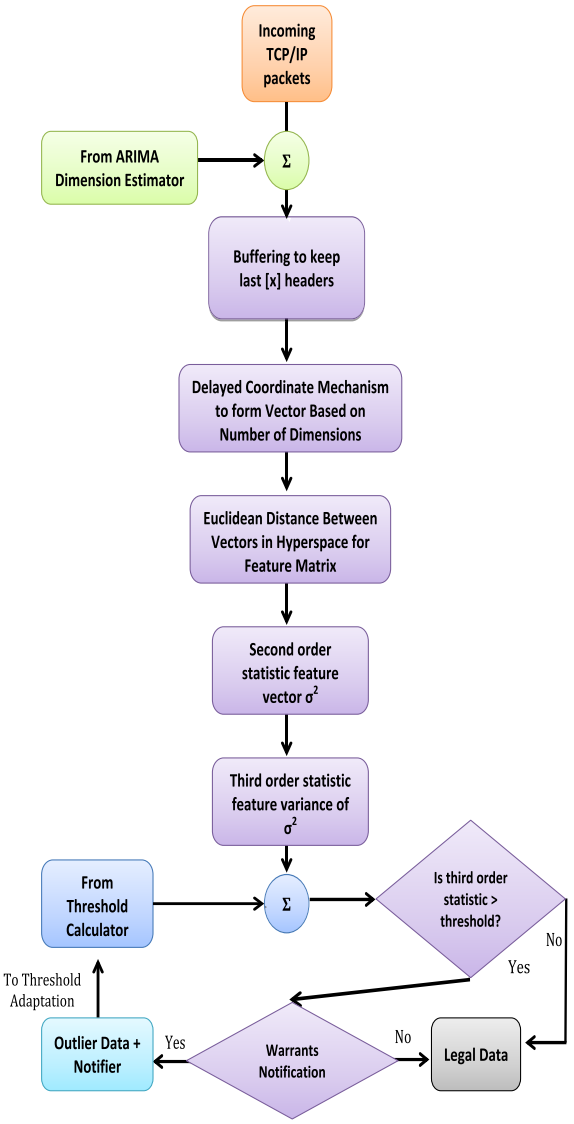


FIGURE 4. Flowchart for quantification of outliers, showing the process followed to decide if a certain packet is marked as legal or illegal.

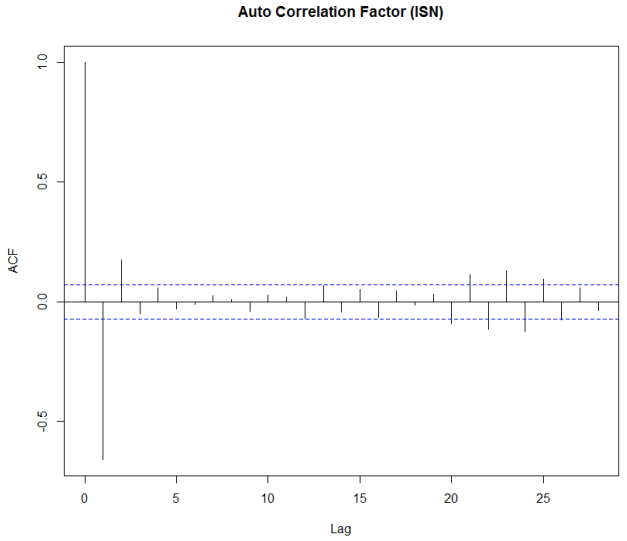


FIGURE 5. Auto-Correlation Function plotting stationary ISNs with delayed coordinate at m = 4, where dataset is a collection of 1000 ISNs.

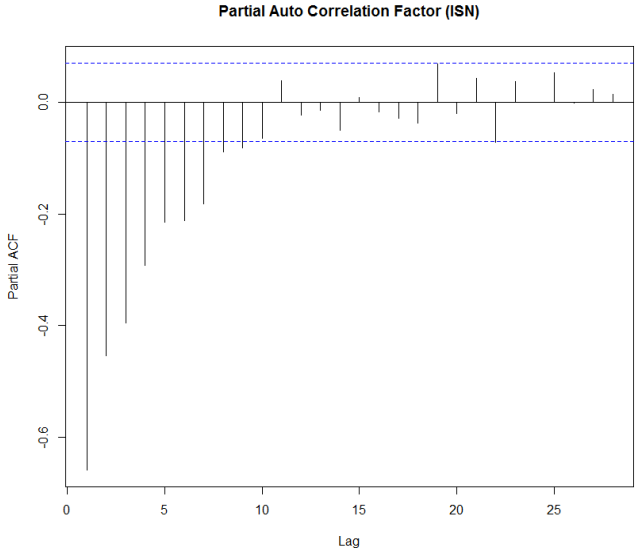


FIGURE 6. Partial Auto-Correlation Function plots for stationary ISN delayed coordinate data at m =4, where dataset is a collection of 1000 ISNs.

```
        Augmented Dickey-Fuller Test

data:  s2
Dickey-Fuller = -19.655, Lag order = 9, p-value = 0.01
alternative hypothesis: stationary
```

FIGURE 7. Dickey-Fuller test for proving that ISN series has been made stationary at m= 4 after applying delayed coordinate method for phase space reconstruction statistical analysis.

was dictated by the specification of covert data creation code called Covert_TCP.

The amount of data collected has direct relation to the accuracy of threshold values. For the purpose of computing a threshold value, the following strategies were applied to ensure that the threshold value obtained had the least amount of skewness and the subset of collected data being processed was not too large. To evaluate this, the following data sizes 100, 350, 700, 1000, 2000 and 3000 were used and the amount of percentage change in the threshold value was measured to ensure durability and precision.

The results from the computation of threshold values using these data sizes are shown in Figure 8. The results of this experiment indicate that using 1000 packets as the data size provides us with the most balanced value with respect to the training period as well as the asymptotic point where the amount of gain achieved by using any higher data size approaches to 0. Hence, for our testbed configuration, the 1000 data size and its corresponding value for the threshold were chosen. Note that this is not constant globally and can change based on hardware metrics available while computing the threshold value.

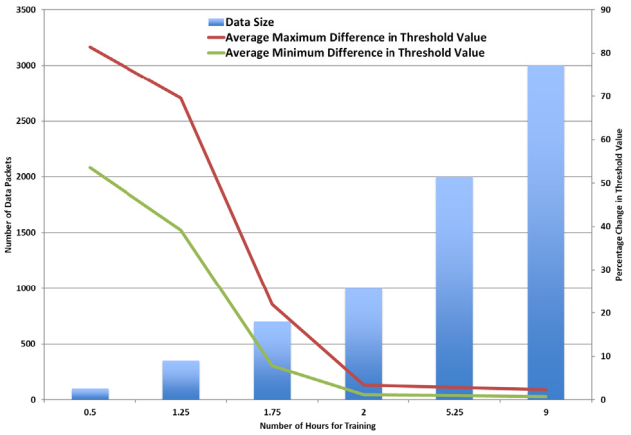


FIGURE 8. Average changes in threshold value with respect to data sizes and time required for creation of training model.

The main objective of the experiments was to re-enforce the threshold model by using it as a benchmark value for testing independent TCP packets in the context of sending information as a covert ISN value in the TCP header.

Experiment 1- The data for this experiment was strategically created in order to have covert sequence numbers in an alternate pattern, whereby the first TCP packet has a true ISN while the second TCP packet has a covert ISN inserted into the sequence number field.

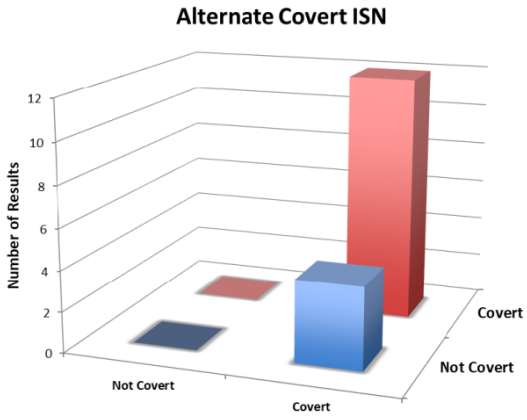
Experiment 2 - The data for this experiment was created in order to have covert sequence numbers appearing together in a bulk of six consecutively placed TCP packets followed by a true sequence number.

Experiment 3 – The data for this experiment has no false covert entries, i.e., all sequence numbers are true.

In experiment 1, it was found that with the alternate pattern of covert ISNs injected into TCP packets, our algorithm successfully located illegitimate entries of covert data in the TCP packets. The Number of True Positives (NTP) was 3 out of 4, while the Number of False Negatives (NFN) was 1 out of 4, causing a Type II error. Figure 9 shows that the Type I error was found to be very low when alternate packets within a group were covert packets.

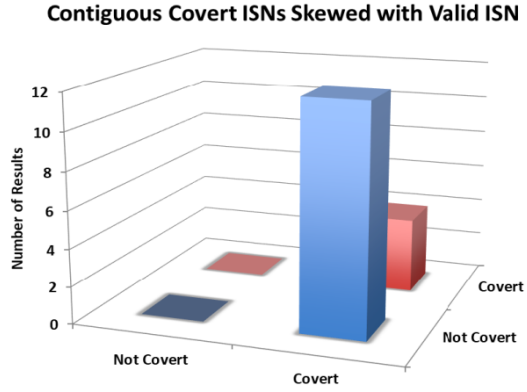
In experiment 2, it was found that with six consecutive entries of covert ISNs injected into TCP packets, our algorithm was not fully successful in locating illegitimate entries of covert data in the TCP packets. NTP was 1 out of 4 while NFN was 3 out of 4, causing a Type II error as well. Figure 10 shows that the Type II error was high as a valid ISN was placed within a group of covert packets. It was noted that the variance got skewed with one true entry in a buffer of four entries. In the cases where all entries were found to be covert, our algorithm was found to be very effective.

In experiment 3, it was found that with all true entries of ISNs in TCP packets, our algorithm was very successful in locating the legitimate entries of covert data in the TCP packets. The Number of True Negatives (NTN) was 4 out of 4. Figure 11 shows that all the valid ISNs when processed through our algorithm were found to be Not Covert.



	Not Covert	Covert
Not Covert	0	4
Covert	0	12

FIGURE 9. Illustration showing that the accuracy of algorithm had a low Type I error for alternate covert ISNs with 50% of packets being covert.



	Not Covert	Covert
Not Covert	0	12
Covert	0	4

FIGURE 10. Illustration showing the accuracy of algorithm with Type I and Type II errors for contiguous covert ISNs with 25% of packets found to be covert.

For the evaluation of the outlier quantification algorithm, the covert data was injected into the ISN field in a TCP packet using a strategy ASCII conversion, while ensuring that covert data was being sent in a continuous stream, an alternate way or with a known gap of valid sequence numbers.

It was observed that Type I errors were low to the amount of 25% for 4 out of 5 experiments, while type II errors were found only in one experiment out of 5. It is noteworthy that these experiments helped with establishing the boundaries of the algorithm. Furthermore, the quantification algorithm was also tested on valid sequence number data in isolation to evaluate errors raised. The algorithm performed as expected in 3 out of 3 instances and did not report any Type I and Type II errors.

C. COMPARISON WITH RELATED WORKS

The research work proposed in [21] and [22] adopts a Support Vector Machine (SVM) technique proved to be highly

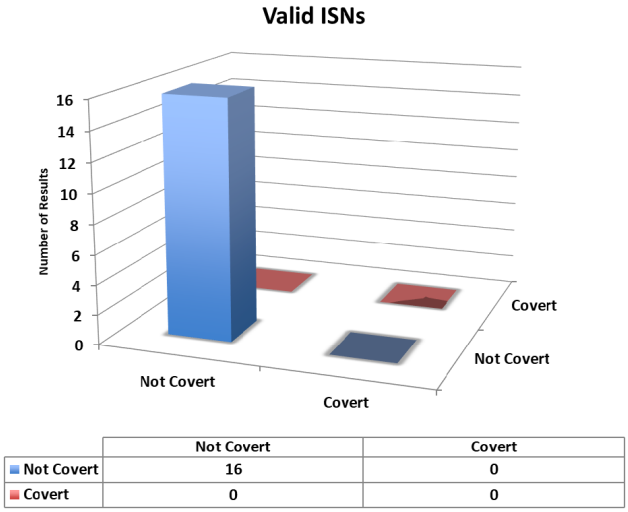


FIGURE 11. Illustration showing that the accuracy of algorithm has no Type I and Type II errors for valid ISNs. This was a test of our algorithm on a legal stream of data. There were no alerts raised.

effective to detect covert channels in comparison with other techniques. As the SVM is the best approach currently used for covert data channel detection, it was an obvious choice for us to compare our proposed algorithm against this type of detection approach in [21] in the context of ISNs. The areas compared include the number of features used, total correctness, detection method and average time to detect as shown in Table 1.

Table 1 shows the results of our proposed detection algorithm against the two different SVM based systems. The results were produced based on our experiments conducted on the Windows 7 operating system where, for all cases, the training data was created using the Covert_TCP tool. The training data and covert data were stored in the operating system for analysis in SVM-cases 1 and 2. For our proposed model, the covert data was sent over the network and analysed by our model on the fly. Table 1 shows that our proposed detection algorithm outperforms the SVM-based approach in each performance metric apart from the number of features in the second SVM case. Our algorithm exploits the fact that ISNs are pseudo random numbers which appear to be chaotic but can be predicted with certain accuracy. This greatly reduces the complexity of our algorithm, which makes it simpler to implement. Note that Table 1 indicates that the accuracy of SVM decreases if non-linear data is used instead of linear chaotic data. The SVM is unable to detect non-linear chaotic data. In order to boost the accuracy of the SVM, the number of features to be considered in a TCP packet needs be increased to three, i.e., the Sequence Number, TCP Control Flag and TCP Checksum fields. The number of features was increased in order to improve the accuracy of the SVM. Hence, the computational complexity of the SVM for the purpose of training is evaluated as O (number of samples x number of features), which includes solving convex optimisation. However, our method only uses a single

TABLE 1. Performance comparison between SVM and our proposed model.

Method	SVM – Case 1	SVM – Case 2	Proposed Detection Model
Number of Features	3	1	1
Training ISN Data	10,000	10,000	1000
Result Accuracy	99%	92%	100%
Detection Method	Offline	Offline	Online
Average Time to Detect (based on encoded data)	35.3s	15.3s	3.03s

feature and does not contain any convex optimization issue. This leads to the computational complexity of the method being O(3× number of samples). This effectively means that our method is able to not only provide higher accuracy with a lower number of features but also offer those results in a fraction of time. By using third order statistical features, our algorithm has identified normal and stego-ISNs with the accuracy rate of 100%. For the Windows 7 operating system, our model needs to be created once with 1000 normal ISNs, while the SVM uses 10,000 ISNs to train the model, including 5000 normal and 5000 stego-ISNs. Here, the stego-ISNs were used in order to simulate existing network conditions where data leakage is carried in chunks, not in a continuous stream. Finally, our method is deployed online for steganalysis, whereas the SVM works only in offline mode. Hence, it is possible that a covert message would pass through before the completion of the SVM model training or steganalysis.

VIII. CONCLUSION

A statistical technique has helped with the quantification of the level of covert communication by computing the deviation of observed data from the expected threshold value. The technique has a two-stage analysis process where a deviation score is calculated in real time over a buffer of collected TCP headers as well as over a larger collection of TCP headers. It offers an insight into temporal data leakage detection for larger data collections, informing an analyst of time based indicative analysis as well as on-the-fly outlier analysis, which are both improvements on existing detection techniques. Furthermore, the proposed framework is novel in that not all deviation scores derived by our decision maker component are flagged as data leakage incidents; instead, the proposed solution offers further analysis using a novel approach to the quantification of skewed results, which informs about the probability of data leakage. The process uses the moving average approach for making decisions about data leakage, thereby increasing the level of accuracy as well as reducing the likelihood of the Type I and Type II errors [9].

REFERENCES

[1] I. Wagner and D. Eckhoff, "Technical privacy metrics: A systematic survey," *ACM Comput. Surv.*, vol. 51, no. 3, p. 57, 2018.

[2] W. Mazurczyk, S. Wendzel, I. A. Villares, and K. Szczypiorski, "On importance of steganographic cost for network steganography," *Secur. Commun. Netw.*, vol. 9, no. 8, pp. 781–790, May 2016.
[3] X. Zhang, Y.-A. Tan, C. Liang, Y. Li, and J. Li, "A covert channel over VoLTE via adjusting silence periods," *IEEE Access*, vol. 6, pp. 9292–9302, 2018.
[4] B. Dimitrova and A. Mileva, "Steganography of hypertext transfer protocol version 2 (HTTP/2)," *J. Comput. Commun.*, vol. 5, no. 5, pp. 98–111, 2017.
[5] S. Wendzel, W. Mazurczyk, and S. Zander, "Unified description for network information hiding methods," *J. Universal Comput. Sci.*, vol. 22, no. 11, pp. 1456–1486, 2016.
[6] X. Yu, Z. Tian, J. Qiu, and F. Jiang, "A data leakage prevention method based on the reduction of confidential and context terms for smart mobile devices," *Wireless Commun. Mobile Comput.*, vol. 2018, pp. 1–11, Oct. 2018.
[7] M. Goodrich and R. Tamassia, *Introduction to Computer Security*. London, U.K.: Pearson, 2018.
[8] M. Wang and L. Tian, "From time series to complex networks: The phase space coarse graining," *Phys. A, Stat. Mech. Appl.*, vol. 461, pp. 456–468, 2016.
[9] S. Abarca, "An analysis of network steganographic malware," Dept. Master Sci. Cybersecur., Utica College, Utica, NY, USA, Tech. Rep., 2018.
[10] H. Zhao and Y.-Q. Shi, "Detecting covert channels in computer networks based on chaos theory," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 2, pp. 273–282, Feb. 2013.
[11] W. Mazurczyk, S. Wendzel, and K. Cabaj, "Towards deriving insights into data hiding methods using pattern-based approach," in *Proc. 13th Int. Conf. Availability, Rel. Secur.*, 2018, p. 10.
[12] S. Taheri, M. Mahdavi, and N. Moghim, "A dynamic timing-storage covert channel in vehicular ad hoc networks," *Telecommun. Syst.*, vol. 69, no. 69, pp. 415–429, 2018.
[13] O. Darwish, A. Al-Fuqaha, G. B. Brahim, I. Jenhani, and A. Vasilakos, "Statistical hierarchical analysis with deep neural network-based framework for covert timing channels detection," *Appl. Soft Comput.*, vol. 82, Jun. 2019, Art. no. 105546.
[14] K. Kaur, I. Gupta, and A. K. Singh, "A comparative study of the approach provided for preventing the data leakage," *Int. J. Netw. Secur. Appl. (IJNSA)*, vol. 9, no. 5, pp. 21–33, 2017.
[15] H. Lema, F. Simba, and A. Ally, "Preventing utilization of shared network resources by detecting IP spoofing attacks through validation of source IP address," in *Proc. IST, Africa Week Conf. (IST-Africa)*, 2018, pp. 1–8.
[16] A. J. Ty, Z. Fang, R. A. Gonzales, P. J. Rozdeba, and H. D. Abarbanel, "Machine learning of time series using time-delay embedding and precision annealing," Feb. 2019, *arXiv:1902.05062*. [Online]. Available: https://arxiv.org/abs/1902.05062
[17] D. Lambic, A. Jankovic, and M. Ahmad, "Security analysis of the efficient chaos pseudo-random number generator applied to video encryption," *J. Electron. Test.*, vol. 34, pp. 709–715, 2018.
[18] S. Sheela and S. Sathyanarayana, "Application of chaos theory in data security-a survey," *ACCENTS Trans. Inf. Secur.*, vol. 2, no. 5, pp. 1–15, 2017.
[19] F. Takens, "Detecting strange attractors in turbulence," in *Dynamical Systems and Turbulence, Warwick* (Lecture Notes in Mathematics), vol. 898. Berlin, Germany: Springer, 1981, pp. 366–381.
[20] H. F. Lopes. *AR, MA and ARMA Models*. Accessed: Sep. 23, 2016. [Online]. Available: http://hedibert.org/wp-content/uploads/2016/04/arma.pdf
[21] M. Stamp, "A survey of machine learning algorithms and their application in information security," in *Guide to Vulnerability Analysis for Computer Networks and Systems*. Cham, Switzerland: Springer, 2018, pp. 33–55.
[22] T. Zseby, F. I. Vázquez, V. Bernhardt, D. Frkat, and R. Annessi, "A network steganography lab on detecting TCP/IP covert channels," *IEEE Trans. Educ.*, vol. 59, no. 3, pp. 224–232, Aug. 2016.

HANAA NAFEA received the master's degree in computer science from Nottingham Trent University, Nottingham, U.K., in 2012. She is currently pursuing the Ph.D. degree with Liverpool John Moores University (LJMU), U.K. She is also a Lecturer with the Department of Computer Science, Taibah University, Almadinah, Saudi Arabia. Her current research interests include network security, the security of complex systems, intrusion detection, secure service composition, privacy-preserving data aggregation, cryptography computer science, and cloud security.

KASHIF KIFAYAT received the Ph.D. degree in cyber security from Liverpool John Moores University, Liverpool, U.K., in 2008. He is currently a Professor and the Chair of the Cyber Security Department, Air University, Islamabad, Pakistan. Prior to this, he was a Reader in cyber security with Liverpool John Moores University. His current research interests include network security, the security of complex systems, intrusion detection, secure service composition, privacy-preserving data aggregation, cryptography, computer forensics, and the IoT security. He has published around 90 articles in international conference proceedings and journals, and served in a number of conferences IPCs and journal editorial boards. He has also played a key role in many funded research and development projects related to his research topics.

QI SHI received the Ph.D. degree in computing from the Dalian University of Technology, China. He then worked as a Research Associate with the University of York, U.K. He then joined Liverpool John Moores University (LJMU), U.K., as a Lecturer and then a Reader before becoming a Professor. He is currently a Professor in computer security and the Director of the PROTECT Research Centre, Department of Computer Science, LJMU. He has many years of research experience in a number of areas, e.g., the IoT security, intrusion detection, secure service composition, privacy-preserving data aggregation, cryptography, computer forensics, formal security models, and cloud security. He has published over 250 articles in international conference proceedings and journals, and served in a number of conference IPCs and journal editorial boards. He has also played a key role in many funded research and development projects related to his research topics.

KASHIF NASEER QURESHI received the Ph.D. degree from the University of Technology Malaysia (UTM), in 2016. He is currently a Senior Assistant Professor with Bahria University, Islamabad, Pakistan. His research interests focus on the Internet of connected Vehicles (IoV), electronic vehicle (EV) charging management planning and recommendation, and the IoT use case implementation in wireless sensor networks, network security and network forensics. He has been involved in a variety of research and development projects, and published widely in various research areas. He has been a Reviewer of a number of reputable academic journals and published around 100 articles. He is currently a Co-PI in the Cyber Reconnaissance and Combat (CRC) Project in Bahria University.

BOB ASKWITH received the Ph.D. degree in network security from Liverpool John Moores University (LJMU), U.K., in 2000. He has been involved in a variety of research and development projects, and published widely in security-related areas. He is the Academic Subject Leader for M.Sc. and B.Sc. programs in cyber security and forensics with the Department of Computer Science, LJMU. His current research interests include network security, network forensics, system-of-systems security, and security education.