Noname manuscript No.
(will be inserted by the editor)

Cyber-Attack Detection via Non-Linear Prediction of IP Addresses: An Innovative Big Data Analytics Approach

Alfredo Cuzzocrea · Edoardo Fadda · Enzo Mumolo

Received: date / Accepted: date

Abstract Computer network systems are often subject to several types of attacks. For example, an excessive traffic load sent to a web server for making it unusable is the main technique introduced by the Distributed Denial of Service (DDoS) attack. A well-known method for detecting attacks consists in analyzing the sequence of source IP addresses for detecting possible anomalies. With the aim of predicting the next IP address, the Probability Density Function of the IP address sequence is estimated. Anomalous requests are detected via predicting source's IP addresses in future accesses to the server. Thus, when an access to the server occurs, the server accepts only the requests from the predicted IP addresses and it blocks all the others. The approaches used to estimate the Probability Density Function of IP addresses range from the sequence of IP addresses seen previously and stored in a database to address clustering, for instance via the K-Means algorithm. Instead, the sequence of IP addresses is considered as a numerical sequence in this paper, and non-linear analysis of this numerical sequence is applied. In particular, we exploited non-linear analysis based on Volterra Kernels and Hammerstein models. The experiments carried out with datasets of source IP address sequences show that the prediction errors obtained with Hammerstein models are smaller than those obtained both with the Volterra Kernels and with the sequence clustering based on the K-Means algorithm.

A. Cuzzocrea
University of Calabria, Rende, Italy
& LORIA, Nancy, France
E-mail: alfredo.cuzzocrea@unical.it

E. Fadda
Politecnico di Torino, Torino, Italy
ISIRES, Torino, Italy
E-mail: edoardo.fadda@polito.it

E. Mumolo
University of Trieste, Trieste, Italy
E-mail: mumolo@units.it

Keywords Cyber-Attack · Distributed Denial of Service · Hammerstein Models.

1 Introduction

User modeling is an important task for web applications dealing with large traffic flows. Predict future situations or classify current states are two possible applications of these techniques. Several papers show the importance of user modeling in other problems such as improving detection and mitigate of Distributed Denial of Service (DDoS) attacks (see [19, 28, 22]), improving the quality of service (see [32]), individuate click fraud detection and optimize traffic management. In peer-to-peer (P2P) overlay networks, IP models can also be used for optimizing request routing [2]. How to manage the actual traffic is a field of application of these techniques. If only one class of users is known another family of methods often used is the detection ones. If, for example, an Intrusion Prevention System wants to mitigate DDoS attacks, the only information that it can use are inferred from the normal traffic before the attack. Thus, the different behavior is the only characteristic that can be used in order to detect outliers. For example, the normal traffic before the attack is the only information that an Intrusion Prevention System wanting to mitigate DDoS attacks can use. In this paper, we deal with the management of DDoS because nowadays it has become a major threat in the internet. A large scaled networks of infected PCs (usually called *bots* or *zombies*) that combine their bandwidth and computational power for overloading a publicly available service and denialing it for legal users are used for those attacks. It is worth noting that all public servers are vulnerable to DDoS attacks due to the open structure of the internet. The bots are usually acquired automatically by hackers by using software tools to scan through the network, detecting vulnerabilities and exploiting the target machines. Due to the current internet infrastructure, the only solution to this problem has proven to be the mitigation of DDoS attacks in the machines near to the target servers. This protection aims to identify malicious requests in order to limit their destabilizing effect on the servers. There are multiple strategies when dealing with DDoS attacks. Near-target filtering solutions has proven to be the most effective ones. These techniques use the data contained in the IP packet header information to estimate normal users behavior. Then, during an attack the accesses from outliers are denied. The IP addresses of the users are the data that all the methods for discrimination of DDoS traffic have in common. Due to the huge IP address space, storing the IP addresses and making inferences on these data are not easy task. In this paper, we present a novel approach based on system identification techniques and, in particular, on the Hammerstein model. The following sections composed the paper. Section 2 presents the state-of-the-art methods for DDoS traffic classification. Since a comprehensive literature review is out of the scope of the present work, we suggest to the reader [16], where a broader overview of state-of-the-art mitigation research is given. In Sections 3 and 4 we present our proposed technique based on the Hammerstein model and we recall some similar models. Section 5 presents the experimental results and confirms the effectiveness of our approach. DDoS mitigation is the most important practical application for IP density estimation. Nevertheless, we present the topic in much general way by not restricting the work to this application. In fact, our generic view on IP density estimation may be valuable to other applications as well. Applications as optimizing routing in peer-to-peer networks, preferring regular customers in overload situations (flash crowd events) and identifying non-regular users on websites during high click rates on online advertisements (click fraud detection), are possible applications of the proposed methodology. In Section 6, we provide some interesting, further remarks that focus the attention on the integration of our proposed framework with emerging big data analytics systems. Finally, Section 7 reports the conclusions and proposes future research lines.

2 Related Work

In this Section, we review existing IP density estimation approaches. A probability density function (PDF) is used to formulate the implicitly used ideas in a probabilistic way. In Figure 1, we report a summary table of the most recent applications.

2.1 History-Based IP Filtering

One of the first work on the field of IP filtering is [29]. An algorithm called History based IP Filtering (HIF) is proposed in their work. During an attack, HIF allows the access to the website only to the IP addresses belonging to a database containing all frequently observed addresses (IAD).

The idea behind this algorithm was introduced in [20]. In this paper, it is observed that the IP addresses from Code Red worm attacks were different from the standard IPs accessing the website. Thus, HIF adds an IP address to the IAD only if a certain threshold is exceeded. This threshold can be measured with different methods, e.g., a certain number of packets, a fixed number of visits in a certain time window, etc. The advantage of the HIF algorithm is the low computational load required for its implementation. A main drawback of this algorithm is that the differentiation between users which revisit the server more often than others is not consider. Therefore, a less precise density estimation is considered. Moreover, if an IP address belongs to the IAD, it is not removed unless specific but fixed rules are applied for the update. This prevent the method to adapt to the dynamic context of the web. A kind of history-based IP filtering approach is introduced in [26]. The focus of the paper is just on DDoS attacks, and the authors present it as one of the most difficult security issues on the Internet today. Indeed, these attacks can without much of a stretch, fumes the assets of the potential victims. Since the aggressors regularly produce their IP delivers to shroud their character, the problem is recognised by the authors to be much more extreme. It is worth noting that the problem in the paper is difficult also because the only guard mechanism against DDoS attacks is to filter the IP addresses to the victim's side. Due to this situation, regardless of whether the attacking traffic is filtered by the victim, the attacker may achieve the objective of blocking access to the victim's bandwidth. To solve this issue, the tracking of the wellspring of an attack is enabled by using IP-Traceback. The authors claim that by using this technique, it is possible to minimize the attack when it is in progress. Due to these characteristics of the problem, authors minimize the quantity of malicious packets entering the network by means of a hybrid method. They also introduce a quantum annealing technique at the server side to identify and mitigate the DDoS attack. Client puzzle is used in order to minimize the attack messages as a part of the ingress router and at the egress side is used the path fingerprint. The authors conclude the paper by presenting simulation studies which prove that recognizing and mitigating the DDoS attacks is optimally done by the proposed techniques.

2.2 Adaptive History-Based IP Filtering

To compensate the shortcomings of HIF, [19] presents the Adaptive History-based IP Filtering (AHIF). The probability that an IP address s is not malicious is estimated by the methods as follows:

$$\phi(s) = \frac{n_s}{\sum_{i=0}^{N-1} n_{s_i}} \tag{1}$$

where n_s is the number of accesses of IP addresses s during the standard period. It is worth noting that HIF just approximate Eq. (1) by binary variables. Eq. (1) can be interpreted as the conditional probability to observe the IP s given that the server is not under attack. A threshold over $f(s)$ is used for predicting the appearance of an IP address (or an IP range). Eq. (2) shows the decision rule r applied by AHIF algorithm:

$$r = \begin{cases} reject \quad if \ \phi(s) \ \geq \ \alpha \\ accept \quad otherwise \end{cases} \tag{2}$$

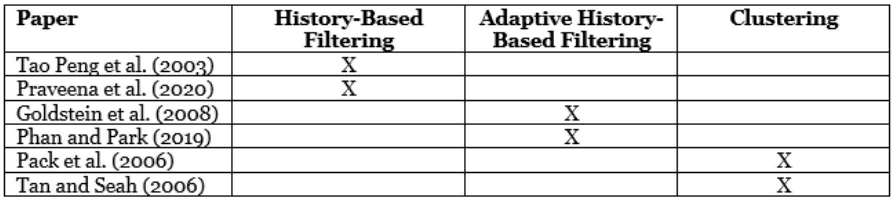
It should be noted that *adaptive metaphors* have been applied in several other domains, with relevant success (e.g., [5,30]).

Given a set of constant width networks with fixed network masks ranging from 16 up to 24 bit, during attack mitigation, the most appropriate network mask is chosen such that a maximum number of firewall rules is not exceeded and the attack traffic is reduced to be below the maximum server capacity. The adaptive method is shown to perform better in terms of predicting user IP addresses during an attack. Nevertheless, the AHIF algorithm does not consider neighbor relations (i.e., relations between source networks).

[25] is in line with the class of adaptive history-based IP filtering approaches. In order to study some DDoS aspects, the specific case of Software-defined networking (SDN)-based Clouds is considered by the author. SDN is the state of the art outcome for transforming the Internet infrastructure to be more programmable, configurable, and manageable. It is the results of extensive research efforts over the past few decades toward. However, critical cyber-threats in the SDN-based cloud environment are rising rapidly, among all cyber-attacks, DDoS attack is one of the most damaging. Therefore, an efficient solution to tackle DDoS attacks in the SDN-based Cloud environment is proposed. The algorithm proposed for improving the traffic classification is composed by a new hybrid machine learning model based on support vector machines and self-organizing map algorithms. Then, they propose an enhanced history-based IP filtering scheme (eHIPF) to improve the attack detection rate and speed. Finally, a novel mechanism that combines both the hybrid machine learning model and the eHIPF scheme is introduced in order to better defend from a DDoS attack in the SDN-based cloud environment.

2.3 Clustering of Source Address Prefixes

In [22], the authors introduce algorithms for mitigating DDoS attacks by filtering source address prefixes. Unlike AHIF, different sizes network are considered. The authors use a hierarchical agglomerative clustering algorithm with single linkage for finding the most appropriate network masks. The used distance measure is defined with respect to network boundaries. In general, both the amount of requests from a source network and the neighboring density estimation (from a generalized clustering method) are taken into account. Unfortunately, the proposed technique is not applicable in practice on large source IP datasets since hierarchical clustering methods consume a lot of memory, (see [28]).



Paper	History-Based Filtering	Adaptive History-Based Filtering	Clustering
Tao Peng et al. (2003)	X		
Praveena et al. (2020)	X		
Goldstein et al. (2008)		X	
Phan and Park (2019)		X	
Pack et al. (2006)			X
Tan and Seah (2006)			X

Fig. 1 Summary of state-of-the-art IP filtering approaches.

3 Non-Linear Analytic Prediction of IP Addresses

Data driven identification of mathematical models of physical systems starts with representing the systems as a black box. In other terms, the internal mechanisms are totally unknown to us despite the availability of inputs and outputs. Thus, choosing the model representative of the system is the first step of identification techniques. The second step is to estimate its parameters through an optimization algorithm so that the model mimics at good as possible the inner mechanisms of the non-linear system. The test of the goodness of fit are performed by using the available inputs and outputs. This approach is, for instance, widely used in the related *big data analytics* area (e.g., [12,8, 10,13,11,14,4,31,1,3]).

In this work, the Linear-In-the-Parameters (LIP) predictors are considered. They belong to the sub-class of non-linear predictors. LIP predictors have many interesting features, the one that is more interesting in our setting is a linear dependence of the predictor output on the predictor coefficients. Such predictors are inherently stable, and their estimation is guaranteed to converge to a globally minimum solution (in contrast to other types of non-linear filters whose cost function may exhibit many local minima). We start by considering a causal, time-invariant, finite-memory, continuous non-linear predictor. Eq. (3) reported its mathematical model:

$$\hat{s}(n) = f(s(n-1), \dots, s(n-N)) \tag{3}$$

where $s(n)$ is the input signal, $\hat{s}(n)$ its estimation and $f(\cdot)$ is a general continuous non-linear function describing the state of the system. We can expand $f(\cdot)$ with a series of basis functions $f_i(\cdot)$, as shown in Eq. (4):

$$\hat{s}(n) = \sum_{i=1}^{\infty} h(i) f_i(s(n-i)) \tag{4}$$

Eq. (4) holds for proper coefficients $h(i)$. As commonly done in the analysis of non linear systems, we truncate the series in Eq. (4) to the first N terms. Thus, we obtain Eq. (5):

$$\hat{s}(n) = \sum_{i=1}^{N} h(i) f_i(s(n-i)) \tag{5}$$

In the general case, a linear-in-the-parameters non-linear predictor is described by the input-output relationship as in Eq. (6):

$$\hat{s}(n) = \vec{H}^T \vec{X}(n) \tag{6}$$

where the row vector containing predictor coefficients is represented by $\vec{H}^T$ and the corresponding column vector whose elements are non-linear combinations of the input samples is represented by $\vec{X}(n)$.

3.1 Linear Predictor

Linear prediction is a well-known technique with a long history [21]. Given a time series $\vec{X}$, the optimal approximation of sample $x(n)$ is obtained from a linear combination of the N most recent samples. Hence, the linear predictor is described in Eq. (7):

$$\hat{x}(n) = \sum_{i=1}^{N} h_1(i)x(n-i) \tag{7}$$

or, in matrix form as in Eq. (8):

$$\hat{x}(n) = \vec{H}^T \vec{X}(n) \tag{8}$$

where Eq. (9) and Eq. (10) define the coefficients of Eq. (8):

$$\vec{H}^T = \begin{bmatrix} h_1(1) \; h_1(2) \, \dots \, h_1(N) \end{bmatrix} \tag{9}$$

$$\vec{X}^T = \begin{bmatrix} x(n-1) \; x(n-2) \, \dots \, x(n-N) \end{bmatrix} \tag{10}$$

3.2 Non-Linear Predictor Based on Volterra Series

As Linear Prediction, Non Linear Prediction is the optimal approximation of the new observation computed by using a non linear combination of the N most recent samples. Volterra series is one of the most popular non-linear predictor (see [24]). We report in Eq. (11) a Volterra predictor based on a Volterra series truncated to the second term:

$$\hat{x}(n) = \sum_{i=1}^{N_1} h_1(i)x(n-i) + \sum_{i=1}^{N_2}\sum_{j=i}^{N_2} h_2(i,j)x(n-i)x(n-j) \tag{11}$$

where the symmetry of the Volterra kernel (the h coefficients) is considered. In matrix terms, Eq. (12) represents the Volterra predictor:

$$\hat{x}(n) = \vec{H}^T \vec{X}(n) \tag{12}$$

where Eq. (13) and Eq. (14) show the coefficient and input vectors of Eq. (12):

$$\vec{H}^T = \begin{bmatrix} h_1(1) & h_1(2) \dots h_1(N) \\ h_2(1,1) & h_2(1,2) \dots h_2(N_2,N_2) \end{bmatrix} \tag{13}$$

$$\vec{X}^T = \begin{bmatrix} x(n-1) & x(n-2) \dots x(n-N_1) \\ x^2(n-1) & x(n-1)x(n-2) \dots x^2(n-N_2) \end{bmatrix} \tag{14}$$

3.3 Non-Linear Predictor Based on Functional Link Artificial Neural Networks (FLANN)

FLANN is a single layer neural network without hidden layer. Function expansion of the input signal exploiting suitable orthogonal polynomials is used to model the non-linear relationships between input and output. The most used choices are trigonometric, Legendre and Chebyshev polynomials.

However, the most frequently used basis function used in FLANN for function expansion are trigonometric polynomials [33]. Eq. (15) reports the FLANN predictor:

$$\hat{x}(n) = \sum_{i=1}^{N} h_1(i)x(n-i) + \sum_{i=1}^{N}\sum_{j=1}^{N} h_2(i,j)\cos[i\pi x(n-j)] + \sum_{i=1}^{N}\sum_{j=1}^{N} h_2(i,j)\sin[i\pi x(n-j)] \quad (15)$$

Using the matrix notation of above, it is possible to notice that also the equation of the Flann predictor (i.e., Eq. (15)) can be represented as follows:

$$\hat{x}(n) = \vec{H}^T \vec{X}(n) \quad (16)$$

where the coefficient and input vectors of FLANN predictors are reported in Eq. (17) and Eq. (18), respectively:

$$\vec{H}^T = \begin{bmatrix} h_1(1) & h_1(2) \ldots h_1(N) \\ h_2(1,1) & h_2(1,2) \ldots h_2(N,N) \\ h_3(1,1) & h_3(1,2) \ldots h_3(N,N) \end{bmatrix} \quad (17)$$

$$\vec{X}^T = \begin{bmatrix} x(n-1) & x(n-2) & \ldots x(n-N) & & \\ \cos[\pi x(n-1)] & \cos[\pi x(n-2)] & \ldots & \ldots & \cos[N_2 \pi x(n-N)] \\ \sin[\pi x(n-1)] & \sin[\pi x(n-2)] & \ldots & \ldots & \sin[N_2 \pi x(n-N)] \end{bmatrix} \quad (18)$$

3.4 Non-Linear Predictors Based on Hammerstein Models

Previous research [7] shown that many real non-linear systems, spanning from electromechanical systems to audio systems, can be modeled using a static non-linearity.

System non-linearities are captured by these models by means of a set of non-linear blocks in series with a linear function. Figure 2 represents the general schema of the model.

One of the most famous non-linear models is the one proposed by Hammerstein, which gains his name. Its front-end is composed by a non-linear function

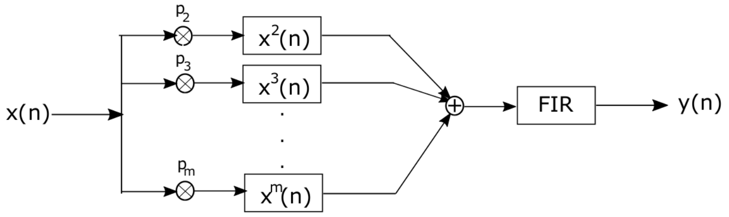


Fig. 2 Representation of the Hammerstein models.

whose inputs are the system inputs. Of course, the type of non-linearity depends on the actual physical system to be modeled. The input to the linear part of the system is the output of the non-linear function.

In the rest of the paper, we use a finite polynomial expansion in order to model non-linearity. Furthermore, we assume that the linear dynamic is realized with a Finite Impulse Response (FIR) filter. Furthermore, in contrast with [7], we assume a mean error analysis and we postpone the analysis in the robust framework in future work. In mathematical terms, Eq. (19) shows the non linear transformation:

$$z(n) = p(1)x(n) + p(2)x^2(n) + p(3)x^3(n) + \dots p(m)x^m(n) = \sum_{i=1}^{M} p(i)x^i(n) \tag{19}$$

The output $z(n)$ is fed to a FIR system as in Eq. (20):

$$\begin{aligned}\hat{x}(n) = h(0) + h(1)z(n-1) + h(2)z(n-2) + \dots + h(N)z(n-N) = \\ = \sum_{j=1}^{N} h(j)z(n-j)\end{aligned} \tag{20}$$

Substituting Eq. (19) in Eq. (20), Eq. (21) is derived:

$$\begin{aligned}\hat{x}(n) = \sum_{j=1}^{N} h(j)z(n-j) = \sum_{j=1}^{N} h(j) \sum_{i=1}^{M} p(i)x^i(n-j) = \\ \sum_{i=2}^{M}\sum_{j=1}^{N} h(j)p(i)x^i(n-j)\end{aligned} \tag{21}$$

Eq. (22) is obtained by Eq. (21) by setting $c(i,j) = h(j)p(i)$:

$$\hat{x}(n) = \sum_{i=1}^{M}\sum_{j=1}^{N} c(i,j)x^i(n-j) \tag{22}$$

and, by writing Eq. (22) in matrix form, we get Eq.

(23):

$$\hat{x}(n) = \vec{H}^T \vec{X}(n) \tag{23}$$

where the matrices $\vec{H}^T$ and $\vec{X}^T$ are defined in Eq. (24) and Eq. (25), respectively:

$$\vec{H}^T = \begin{bmatrix} c(2,1) & c(2,2) & \dots & c(2,N) \\ c(3,1) & c(3,2) & \dots & c(3,N) \\ \dots & \dots & \dots & \dots \\ c(M,1) & c(M,2) & \dots & c(M,N) \end{bmatrix} \tag{24}$$

$$\vec{X}^T = \begin{bmatrix} x^2(n-1) & x^2(n-2) & \dots & x^2(n-N) \\ x^3(n-1) & x^3(n-2) & \dots & x^3(n-N) \\ \dots & \dots & \dots & \dots \\ x^M(n-1) & x^M(n-2) & \dots & x^M(n-N) \end{bmatrix} \tag{25}$$

4 Predictor Parameters Estimation

So far, we saw that all the predictors can be expressed, at time instant n, as follows:

$$\hat{x}(n) = \vec{H}^T \vec{X}(n) \tag{26}$$

where $\vec{X}(n)$ is the input vector (for the aforementioned different definitions), and $\vec{H}^T$ is the parameters vector. For the estimation of the parameters there are several possibilities. Before to analyse them, we generalize the model of Eq. (26). Thus, in the following we consider the general signal $\hat{y}(n)$, in relation to generic input that can be its previous observations.

The general Eq. (27) is then considered instead of Eq. (26):

$$\hat{y}(n) = \vec{H}^T \vec{X}(n) \tag{27}$$

We recall to the reader, that in our application domain, the i-th component of vector $\hat{y}(n)$ represents the number of access from IP address s_i. In the following section, we present two methods for estimating the matrix $\vec{H}^T$. In particular, a block based approach is considered in Section 4.1, while an adaptive approach is considered in Section 4.2.

4.1 Block-Based Approach

The Least Square estimation is based on the minimization of the mathematical expectation of the squared prediction error $e(n) = y(n) - \hat{y}(n)$ shown in Eq. (28):

$$E[e^2] = E[(y(n) - \hat{y}(n))^2] = E[(y(n) - \vec{H}^T \vec{X}(n))^2] \tag{28}$$

Due to the convexity of function in Eq. (28), setting to zero the Laplacian allows us to compute the parameters leading to the minimum estimation error:

$$\nabla_H E[e^2] = E[\nabla_H e^2] = E[2e(n)\nabla_H e] = 0 \tag{29}$$

Thus, Eq. (29) can be written. Its solution is shown in Eq. (30):

$$\vec{H}_{opt} = \vec{R}_{xx}^{-1}\vec{R}_{yx} \tag{30}$$

where the the definition of the statistical auto-correlation matrix of the input vector $\vec{X}(n)$, $\vec{R}_{xx}(n)$, is shown in Eq. (31):

$$\vec{R}_{xx}(n) = E[\vec{X}(n)\vec{X}^T(n)] \tag{31}$$

Instead, Eq. (32) shows the statistical cross-correlation vector between the signal $s(n)$ and the input vector $\vec{X}(n)$:

$$\vec{R}_{yx}(n) = E[y(n)\vec{X}(n)] \tag{32}$$

Eq. (33) and Eq. (34) show the expressions used to estimate the mathematical expectations of the auto and cross correlation, respectively:

$$\vec{R}_{xx}(n) = \frac{\sum_{k=1}^{n} \vec{X}(n)\vec{X}^T(n)}{n} \tag{33}$$

$$\vec{R}_{yx}(n) = \frac{\sum_{k=1}^{n} y(k)(n)\vec{X}(n)}{n} \tag{34}$$

4.2 Adaptive Approach

Let us consider a general second order terms of a Volterra predictor as in Eq. (35):

$$y(n) = \sum_{k=0}^{N-1}\sum_{r=0}^{N-1} h_2(k,r)x(n-k)x(n-r) \tag{35}$$

The generalization of Eq. (35) for higher order term is reported in Eq. (36):

$$y(n) = \sum_{k_1=1}^{N} \cdots \sum_{k_p=1}^{N} c_{k_1} \cdots c_{k_p} H_p\big[x_{k_1}(n), \cdots x_{k_p}(n)\big] \tag{36}$$

In the following, we consider a Volterra predictor based on a Volterra series truncated to the second term as in Eq. (37). It is worth noting that despite the easier notation the general calculation does not change is higher order approximations are considered:

$$\hat{y}(n) = \sum_{i=1}^{N_1} h_1(i)y(n-i) + \sum_{i=1}^{N_2}\sum_{j=i}^{N_2} h_2(i,j)y(n-i)y(n-j) \tag{37}$$

by defining $H^T(n)$ as in Eq. (38), and $X^T(n)$ as in Eq. (39):

$$H^T(n) = |h_1(1), \cdots, h_1(N_1), h_2(1,1), \cdots, h_2(N_2, N_2)| \tag{38}$$

$$\begin{array}{c} X^T(n) = |y(n-1), \cdots, y(n-N_1), y^2(n-1) \\ y(n-1)y(n-2), \cdots, y^2(n-N_2)| \end{array} \tag{39}$$

Eq. (40) is obtained by rearranging Eq. (37):

$$\hat{y}(n) = H^T(n)X(n) \tag{40}$$

We consider the loss function in Eq. (41) in order to estimate the best parameter H:

$$J_n(H) = \sum_{k=0}^{n} \lambda^{n-k} \left[\hat{y}(k) - H^T(n)X(k)\right]^2 \tag{41}$$

where λ^{n-k} weights the relative importance of each squared error. As the reader can notice, Eq. (41) and Eq. (28) are similar. By imposing the gradient of Eq. (41) to zero, we find the H that minimizes the convex function $J_n(H)$. The mathematical expression is reported in Eq. (42):

$$\nabla_H J_n(H) = 0 \tag{42}$$

Eq. (42) is equivalent to Eq. (43):

$$R_{XX}(n)H(n) = R_{yX}(n) \tag{43}$$

Eq. (44) defines $R_{XX}(n)$:

$$\begin{array}{c} R_{XX}(n) = \sum_{k=0}^{n} \lambda^{n-k} X(k)X^T(k) \\ R_{yX}(n) = \sum_{k=0}^{n} \lambda^{n-k} y(k)X(k) \end{array} \tag{44}$$

Eq. (45) shows how the best H can be computed:

$$H(n) = R_{XX}^{-1}(n)R_{yX}(n) \tag{45}$$

It is worth to notice that $R_{XX}(n)$ is a combination of $R_{XX}(n-1)$ and $X(n)X^T(n)$ as in Eq. (46):

$$R_{XX}(n) = \lambda R_{XX}(n-1) + X(n)X^T(n) \tag{46}$$

thus, Eq. (47) can be obtained:

$$R_{XX}^{-1}(n) = \frac{1}{\lambda}\left[R_{XX}^{-1}(n-1) - k(n)X^T(n)R_{XX}^{-1}(n-1)\right] \tag{47}$$

The parameters $k(n)$ are defined in Eq.(48):

$$k(n) = \frac{R_{XX}^{-1}(n-1)X(n)}{\lambda + X^T(n)R_{XX}^{-1}(n-1)X(n)} \tag{48}$$

Analogously to matrix $R_{XX}(n)$, matrix $R_{yX}(n)$ in Eq. (45) can be written as in Eq.(49):

$$R_{yX}(n) = \lambda R_{yX}(n-1) + y(n)X(n) \tag{49}$$

Thus, Eq. (50) is obtained by plugging Eq. (49) and Eq. (47) in Eq. (45) and rearranging the terms:

$$H(n) = H(n-1) + R_{XX}^{-1}(n)X(n)\epsilon(n) \tag{50}$$

Eq. (51) defines ϵ:

$$\epsilon = \hat{y}(n) - H^T(n-1)X(n) \tag{51}$$

by recalling Eq. (48), Eq. (50) can be rewritten as Eq. (52):

$$H(n) = H(n-1) + \epsilon(n)k(n) \tag{52}$$

Let us define matrix $F(n)$ as in Eq. (53):

$$F(n) = S^T(n-1)X(n) \tag{53}$$

Eq. (54) resumes the previous equations in a system:

$$\begin{cases} L(n) = S(n-1)F(n) \\ \beta(n) = \lambda + F^T(n)F(n) \\ \alpha(n) = \frac{1}{\beta(n)+\sqrt{\lambda\beta(n)}} \\ S(n) = \frac{1}{\sqrt{\lambda}}\left[S(n-1) - \alpha(n)L(n)F^T(n)\right] \\ \epsilon(n) = \hat{r}(n-1) - \alpha(n)L(n)F^T(n) \\ \epsilon(n) = H(n-1) + L(n)\frac{\epsilon(n)}{\beta(n)} \end{cases} \tag{54}$$

It should be noted that by solving Eq. (54) the estimation adapts in each step to decrease the error. Thus, a structure similar to the the Kalman filter can be observed. From the computational point of view, the solution of Eq. (40) requires a number of operation that is equal to:

$$6N_{\text{tot}} + 2N_{\text{tot}}^2 \tag{55}$$

where N_{tot} is equal to:

$$N_{\text{tot}} = N_1 + N_2\left(N_2+1\right)/2 \tag{56}$$

5 Experiments

In order to prove the effectiveness of the proposed approach, in this Section we present our experimental results for a simulated dataset and a real one. In this Section, a comparison of the two approaches (namely the prediction based on Volterra systems and the one based on the Hammerstein one) is presented. The following method has been used to generate the simulated dataset:

1. generate a set of $N = 1000$ IP addresses in which the 10% are malicious users. For each of them a normal random variable generates the number of access. In particular the distribution chosen is $\mathcal{N}(\mu, \mu/5)$ where $\mu = \mathcal{U}[10, 100]$ if the IP address is not malicious, $\mu = \mathcal{U}[200, 1000]$ if the user is malicious;
2. during each time step t:
   - pick a random number using a Bernoulli with probability $p = 0.001$, i.e., $X_t \sim \mathcal{B}(0.001)$;
   - an attack is performed if $X_t = 1$, (i.e., one or more malicious users have been selected) else no attach is performed.

In each time step, the algorithm compares the access observed with the one predicted and blocks the access to the website to all those IPs which number of accesses is greater than the forecasted number of accesses. In the real setting, it would be better to improve the robustness of the algorithm by adding a parameter that increase by $10\% - 20\%$ the forecasted number of accesses. Nevertheless, the results of the plain algorithm are presented. It is worth noting that those results are a lower bound of the performance of the algorithm in the real settings. The first analysis that we perform is related to the number of IP address that are not blocked despite being not malicious. We define e_n as in Eq. (57):

$$e_n = \frac{\textit{number of legal IPs blocked}}{\textit{total number of accesses in t}} \quad \forall t = 0, \ldots, T \tag{57}$$

Instead, Eq. (58) defines e:

$$e = \max_{n \in [0,T]} e_n \tag{58}$$

The value e is the maximum percentage of not malicious IP addresses for which the identification algorithm has denied the access in the whole time horizon $[0, T]$. This is an important indicator since it provides a measure of the damage that adopting this technique can generate to standard users. Figure 3 shows the values of e versus different values of probability of attack p. It is worth noting that e is almost negligible for extreme values of p, i.e., p near to 0 and 1. In particular, if $p = 0$, (i.e., no attacks) the server allows the service to all the IP addresses. Instead, if $p = 1$, (i.e., in each time step there is an attack), the individuation of the malicious IP addresses is well performed by both the algorithms. As the reader can notice, the worst values of e are achieved for p near to 0.5. This is reasonable, since $p = 0.5$ is the value characterized by the less information about the probability of an attack. The two values $e = 4\%$ and $e = 7\%$ are the worst performance of the Hammerstein and Volterra model, respectively. These values are reasonable for a real application. The goodness of these results is also strengthened by observing that it is unlucky that the real setting will present a probability of attack greater than few percentage points Thus, the presented results and Figure 3, must be interpreted as an upper bound on the real error e.

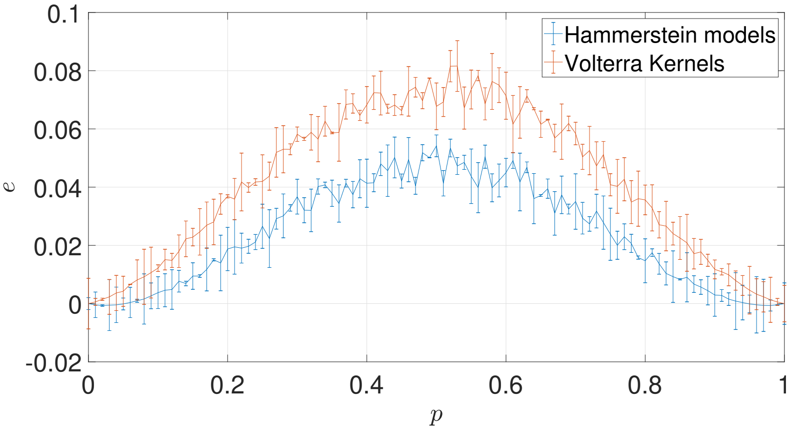


Fig. 3 Maximum error e for different values of probability of attack p.

As the reader can notice, the number of coefficients needed by the Volterra model is greater than the one needed by the Hammerstein (see Eq. (21) with respect to Eq. (11)). Thus, outfitting is probably the cause of the worst performance of the Volterra model in Figure 3. Furthermore, by observing that the Hammerstein model required less coefficients than the Volterra one, we can conclude that also from a computational point of view to use the usage of the Hammerstein model is preferable.

The requests made to the 1998 World Cup Web site between April 30, 1998 and July 26, 1998 [1] are considered as the real data set. During this period of time the database collected 1,352,804,107 requests. The fields of the request structure contain the following information:

- *timestamp* is the time of the request, it is recorded in number of seconds and it has been converted to GMT for improving readability. During the World Cup the local time was 2 hours ahead of GMT (+0200). This amount must be used to adjust the timestamp in order to determine the local time.
- *clientID* is an integer number which identifies the client that issued the request. For privacy reasons the real IP addresses cannot be released. It worth noting that each IP address is mapped though a bijective function

[1] `ftp://ita.ee.lbl.gov/html/contrib/WorldCup.html`

in the *clientID* space. Thus, if IP address s is mapped to *clientID* X on day t then any request in any of the data sets containing. *clientID* X also came from IP address s.

- *objectID* a unique integer identifier for the requested URL; these mappings are also 1-to-1 and are preserved across the entire data set.
- *size* records the number of bytes in the response.
- *method* is the REQUEST method of the client's request (e.g., GET, POST, etc.).
- *status* this field contains two pieces of information; the 2 highest order bits contain the HTTP version indicated in the client's request (e.g., HTTP/1.0); the remaining 6 bits indicate the response status code (e.g., 200 OK).
- *type* if the request required a file, it is the type of file. It is generally the file extension (e.g., .html), or the presence of a parameter list.
- *server* indicates the IP address of the server which handled the request. The upper 3 bits indicate which region the server was at; the remaining bits indicate which server at the site handled the request.

The dataset collects data about the access for 87 days. We split the data in two parts: the first one is used for initializing the estimator in Eq. (37), while the second one is used as test set by using a rolling horizon method (as in [7]). Particularly, for each day t we compute the estimation by using all the IP observations in the previous days $[0, t-1]$. From the computational point of view the more days are considered, the more computational time is required by the algorithms. Thus, as above described the algorithms cannot be applied in the real field without the definition of a fixed time window that limits the number of data considered. Nevertheless, their application of an ad-hoc website for a event with a finite duration is reasonable. Figure 4 shows the results for the Hammerstein and Volterra models.

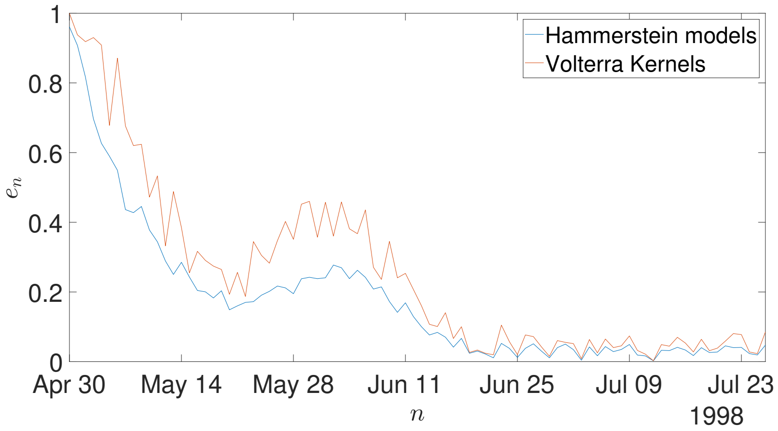


Fig. 4 Estimation error for each day of activity of the website.

Due to the sudden increment of different IP addresses accessing the website for the starting matches of the competition, both methods shown an increment in the estimation error in June. Then, the estimation error decrease exponentially despite dealing with several millions of IPs. The smallest estimation error and the lowest error variance is shown by the Hammerstein model. Furthermore, the amplitude of the error due to the sudden increase of the IPs is bigger for the Volterra model.

Since the computation of the optimal coefficient $H(n)$ may require some time, we measure the percentage of available data that our approach needs in order to provide good results. The average estimation error of each model at time step n, given a subset of the IPs observed in interval $[0, n-1]$ is considered in this experiment. We consider these subsets to be composed of a percentage $ip_{\%}$ of the IP addresses. Figure 5 shows the experimental results on real data.

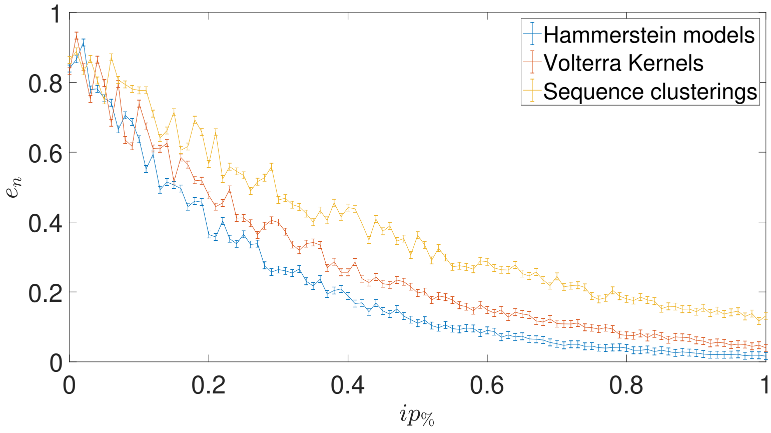


Fig. 5 Estimation error vs. percentage size of the training set.

In this experiment, we also report the performance of the state of the art techniques, in particular we consider the clustering technique proposed in [22].

Both the Volterra kernels and the clustering techniques are outperformed by the Hammerstein models. In particular, the worst performing technique is the clustering one. This is due to the nature of the clustering techniques that exploit the geometric information of the data more than their time dependency. We highlight that the real time applicability of the method is not affected by the computational evaluation of $H(n)$ since the decision of the IP address to block is taken by considering the estimator $\hat{y}(n)$ that is computed once per day. Therefore, the computation of $H(n)$ does not need to be performed in a short amount of time. It is worth noting that the average estimation error is the right point of the graph in Figure 5.

To avoid over-fitting, the best values of $ip_{\%}$ to be considered is $ip_{\%} \sim 60\%$. For further testing the proposed methodologies, we test them using the afore-

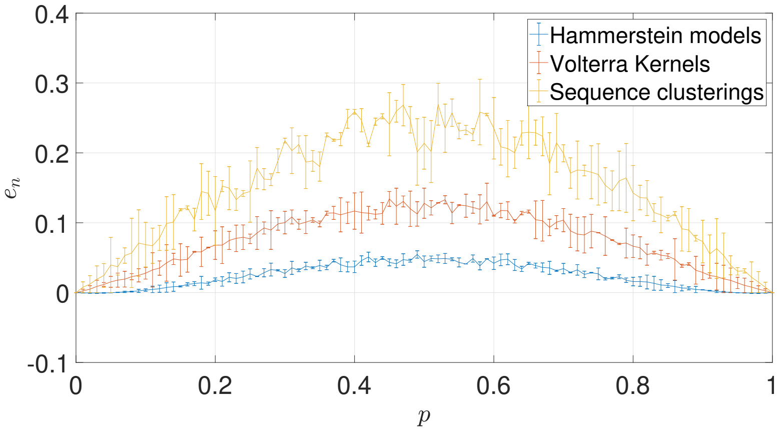


Fig. 6 Estimation error e_n vs. probability of attack p.

mentioned simulation setting using different probabilities of attack. In this way, we can see how the algorithms behave in different conditions. The results are shown in Figure 6. As the reader can notice, the results of the analysis gather from Figure 5 are confirmed. The best models are the Hammerstein ones that achieve the maximum error $e_n \sim 5\%$. These results are particularly important if we consider that the worst performance is achieved for a probability of attack p near 50% that it very unlucky in the real world. The second best models are the Volterra kernels (the worst performance is $e_n \sim 10\%$) while the worst performance are achieved by the Sequence clustering techniques based on K-Means ($e_n \sim 25\%$). We have selected K-Means as the basis of our sequence clustering as it has proven to be very good on this specific applicative setting (e.g., [23]). The interesting fact is that the behaviour of the models is well define and it has statistically significance due to the low standard deviations of the observations. It is worthwhile noting that the standard deviations of the Hammerstein models are smaller than the one of the Volterra kernels that are smaller than the ones of the Sequence clustering techniques. This better result of Hammerstein models is due to the numerical stability of the method. Instead, the sequence clustering techniques present the biggest standard deviations due to their intrinsic stochasticity.

6 Further Remarks: Integration with Big Data Analytics Systems

Recent studies, such as [27], have clearly shown the strong requirement of integrating cyber-security frameworks, like the one we presented on our paper, with innovative big data analytics systems. Our technique, being intrinsically a big data analytics technique, is immediately prone to adhere to this nice integration.

Indeed, when the input dataset to be processed become excessive large, i.e. it takes the form of a big data repository, *scalability* becomes a critical requirement towards the reliability of the overall cyber-security framework. This way, cyber-security techniques must be seamlessly integrated with state-of-the-art big data processing platforms, such as *Hadoop*, *Spark* and so forth, in order to obtain truly-scalable analytics systems over big data repositories (just like collections of IP addresses deriving from a portion of the Web).

Therefore, it follows that the new challenge for next-generation cyber-security frameworks is represented by the strict integration with big data analytics systems, which more and more play the role of enabling technologies for a plethora of emerging applications like smart cities, social networks, bio-informatics, and so forth.

Finally, as regards the integration with big data analytics systems, the following critical aspects must be taken into account:

1. *scalability*, which is the attitude to scale over growing-in-size big data repositories of IPs – this becomes critical in modern applicative settings, perhaps mapped on top of multi-Cloud architectures;
2. *high-rate streaming sources*: the proposed framework takes as input IPs, a well-known class of streaming sources; on the other hand, modern applications demand for *high-rate* IPs, i.e. IPs arriving at a very high inter-arrival time – this puts the basis for specialized acquisition modules that should interface the target data sources and the proposed prediction framework;
3. *heterogeneity*, which is the capabilities of dealing with the variety of IP addresses that can reach very high values – this is a special feature to be taken into account, due to the nature of different domains that characterize popular applicative settings where our framework is supposed to operate (e.g., heterogeneous IP multi-domains);
4. *outlier and false positive management*: outliers and false positives still occur in real-life IP-based application scenarios; this feature poses several challenges to deal-with, due to the reduction of the prediction accuracy that it may derive from this phenomenon.

7 Conclusions and Future Work

In this paper, we presented a new way to deal with cyber-attack by using Hammerstein models. The effectiveness of the proposed techniques is proven by means of experiments using also a real data. Moreover, they have proven to outperforming other well-known techniques (such as clustering). Summarizing, in this paper we made the following contributions:

1. we focus on the problem of detecting DDoS attacks for analyzing and mining sequences of IP addresses;
2. our method is a kind of anomaly detection mining method;
3. our goal is that of predicting the next IP address via analyzing the Probability Density Function of the actual IP address sequence;

4. the core of our approach consists in considering the sequence of IP addresses as a numerical sequence in, and non-linear analysis of this sequence is applied;
5. we exploit non-linear analysis based on Volterra Kernels and Hammerstein models;
6. experiments show that the prediction errors obtained with Hammerstein models are smaller than those obtained both with the Volterra Kernels and with the sequence clustering based on the K-Means algorithm.

Our approach has also some open issues and limitations. For instance, we still need to study how our framework can scale on big data settings, managing billions and billions of IP addresses. Furthermore, we still need to improve our prediction technique as to make it robust with respect to the presence of "false" outliers in data. The latter problem can, in fact, be witness of more sophisticated cyber-attacks.

We propose two future lines of research. First, we want to consider the problem in a stochastic optimization settings, as for example in [18] and [17]. Second, we think that the exploitation of *knowledge management methodologies* can be beneficial for the proposed algorithms (e.g., [9, 6, 15]).

Acknowledgements

This research has been partially supported by the French PIA project "Lorraine Université d'Excellence", reference ANR-15-IDEX-04-LUE.

References

1. Abid, A., Jemili, F.: Intrusion detection based on graph oriented big data analytics. In: M. Cristani, C. Toro, C. Zanni-Merk, R.J. Howlett, L.C. Jain (eds.) Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 24th International Conference KES-2020, Virtual Event, 16-18 September 2020, *Procedia Computer Science*, vol. 176, pp. 572–581. Elsevier (2020)
2. Agrawal, A., Casanova, H.: Clustering hosts in P2P and global computing platforms. In: 3rd IEEE International Symposium on Cluster Computing and the Grid (CCGrid 2003), 12-15 May 2003, Tokyo, Japan, pp. 367–373. IEEE Computer Society (2003)
3. Amen, B., Antoniou, G.: A theoretical study of anomaly detection in big data distributed static and stream analytics. In: 20th IEEE International Conference on High Performance Computing and Communications; 16th IEEE International Conference on Smart City; 4th IEEE International Conference on Data Science and Systems, HPCC/SmartCity/DSS 2018, Exeter, United Kingdom, June 28-30, 2018, pp. 1177–1182. IEEE (2018)
4. Bonifati, A., Cuzzocrea, A.: Storing and retrieving xpath fragments in structured P2P networks. Data Knowl. Eng. **59**(2), 247–269 (2006)
5. Cannataro, M., Cuzzocrea, A., Pugliese, A.: Xahm: an adaptive hypermedia model based on xml. In: Proceedings of the 14th international conference on Software engineering and knowledge engineering, pp. 627–634 (2002)
6. Casas, P., Fiadino, P., Bär, A.: IP mining: Extracting knowledge from the dynamics of the internet addressing space. In: 25th International Teletraffic Congress, ITC 2013, Shanghai, China, September 10-12, 2013, pp. 1–9. IEEE (2013)

7. Cerone, V., Fadda, E., Regruto, D.: A robust optimization approach to kernel-based nonparametric error-in-variables identification in the presence of bounded noise. In: 2017 American Control Conference (ACC). IEEE (2017). DOI 10.23919/acc.2017.7963056
8. Chatzimilioudis, G., Cuzzocrea, A., Gunopulos, D., Mamoulis, N.: A novel distributed framework for optimizing query routing trees in wireless sensor networks via optimal operator placement. J. Comput. Syst. Sci. **79**(3), 349–368 (2013)
9. Cuzzocrea, A.: Combining multidimensional user models and knowledge representation and management techniques for making web services knowledge-aware. Web Intelligence and Agent Systems **4**(3), 289–312 (2006)
10. Cuzzocrea, A., Bertino, E.: Privacy preserving OLAP over distributed XML data: A theoretically-sound secure-multiparty-computation approach. J. Comput. Syst. Sci. **77**(6), 965–987 (2011)
11. Cuzzocrea, A., Maio, C.D., Fenza, G., Loia, V., Parente, M.: OLAP analysis of multidimensional tweet streams for supporting advanced analytics. In: S. Ossowski (ed.) Proceedings of the 31st Annual ACM Symposium on Applied Computing, Pisa, Italy, April 4-8, 2016, pp. 992–999. ACM (2016)
12. Cuzzocrea, A., Moussa, R., Xu, G.: Olap*: Effectively and efficiently supporting parallel OLAP over big data. In: Model and Data Engineering - Third International Conference, MEDI 2013, Amantea, Italy, September 25-27, 2013. Proceedings, pp. 38–49 (2013)
13. Cuzzocrea, A., Russo, V.: Privacy preserving OLAP and OLAP security. In: Encyclopedia of Data Warehousing and Mining, Second Edition (4 Volumes), pp. 1575–1581 (2009)
14. Cuzzocrea, A., Song, I.: Big graph analytics: The state of the art and future research agenda. In: I. Song, A. Simitsis, A. Cuzzocrea (eds.) Proceedings of the 17th International Workshop on Data Warehousing and OLAP, DOLAP 2014, Shanghai, China, November 3-7, 2014, pp. 99–101. ACM (2014)
15. Dehghani, R., Ramsin, R.: Methodologies for developing knowledge management systems: an evaluation framework. J. Knowl. Manag. **19**(4), 682–710 (2015)
16. Dietrich, S., Long, N., Dittrich, D.: Analyzing distributed denial of service tools: The shaft case. pp. 329–339 (2000)
17. Fadda, E., Perboli, G., Squillero, G.: Adaptive batteries exploiting on-line steady-state evolution strategy. pp. 329–341. Springer International Publishing, Cham (2017)
18. Fadda, E., Perboli, G., Tadei, R.: A progressive hedging method for the optimization of social engagement and opportunistic iot problems. European Journal of Operational Research **277**(2), 643 – 652 (2019). DOI https://doi.org/10.1016/j.ejor.2019.02.052
19. Goldstein, M., Lampert, C., Reif, M., Stahl, A., Breuel, T.: Bayes optimal ddos mitigation by adaptive history-based ip filtering. In: Seventh International Conference on Networking (ICN 2008), pp. 174–179 (2008)
20. Jung, J., Krishnamurthy, B., Rabinovich, M.: Flash crowds and denial of service attacks: characterization and implications for cdns and web sites. pp. 293–304 (2002). DOI 10.1145/511446.511485
21. Makhoul, J.: Linear prediction: A tutorial review. Proceedings of the IEEE **63**(4), 561–580 (1975)
22. Pack, G., Yoon, J., Collins, E., Estan, C.: On filtering of ddos attacks based on source address prefixes. pp. 1–12 (2006). DOI 10.1109/SECCOMW.2006.359537
23. Park, S., Suresh, N.C., Jeong, B.: Sequence-based clustering for web usage mining: A new experimental framework and ann-enhanced k-means algorithm. Data Knowl. Eng. **65**(3), 512–543 (2008)
24. Peng, Z., Changming, C.: Volterra series theory: A state-of-the-art review. Chinese Science Bulletin (Chinese Version) **60**, 1874 (2015). DOI 10.1360/N972014-01056
25. Phan, T.V., Park, M.: Efficient distributed denial-of-service attack defense in sdn-based cloud. IEEE Access **7**, 18701–18714 (2019)
26. Praveena, V., Karthik, S., Jeon, G.: Hybrid approach for IP traceback analysis in wireless networks. Wirel. Pers. Commun. **113**(1), 669–690 (2020)
27. Sarker, I.H., Kayes, A.S.M., Badsha, S., AlQahtani, H., Watters, P.A., Ng, A.: Cybersecurity data science: an overview from machine learning perspective. J. Big Data **7**(1), 41 (2020)
28. Tan, H.X., Seah, W.: Framework for statistical filtering against ddos attacks in manets. pp. 8 pp.– (2006). DOI 10.1109/ICESS.2005.57

29. Tao Peng, Leckie, C., Ramamohanarao, K.: Protection from distributed denial of service attacks using history-based ip filtering. In: IEEE International Conference on Communications, 2003. ICC '03., vol. 1, pp. 482–486 vol.1 (2003)
30. Toth, P.: Adaptive online learning environment and web usage mining. In: IEEE 8th International Symposium on Applied Computational Intelligence and Informatics, SACI 2013, Timisoara, Romania, May 23-25, 2013, pp. 61–66. IEEE (2013)
31. Wang, L., Jones, R.: Big data analytics in cybersecurity: Network data and intrusion prediction. In: 10th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference, UEMCON 2019, New York City, NY, USA, October 10-12, 2019, pp. 105–111. IEEE (2019)
32. Yang, Y., Lung, C.H.: The role of traffic forecasting in qos routing - a case study of time-dependent routing. pp. 224 – 228 Vol. 1 (2005). DOI 10.1109/ICC.2005.1494351
33. Zhao, H., Zhang, J.: Adaptively combined fir and functional link artificial neural network equalizer for nonlinear communication channel. IEEE Transactions on Neural Networks **20**(4), 665–674 (2009)