Hindawi
Security and Communication Networks
Volume 2021, Article ID 5542919, 17 pages
https://doi.org/10.1155/2021/5542919

Research Article

An Adaptive Protection of Flooding Attacks Model for Complex Network Environments

Bashar Ahmad Khalaf,[1,2] Salama A. Mostafa,[1] Aida Mustapha,[1] Mazin Abed Mohammed,[3] Moamin A. Mahmoud,[4] Bander Ali Saleh Al-Rimy,[5] Shukor Abd Razak,[5] Mohamed Elhoseny,[6] and Adam Marks[7]

[1]*Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia, Johor 86400, Malaysia*
[2]*Bilad Alrafidain University College, Ba'aqubah 32001, Diyala, Iraq*
[3]*College of Computer Science and Information Technology, University of Anbar, Ramadi 31001, Iraq*
[4]*College of Computer Science and Informatics, Universiti Tenaga Nasional, Kajang, Selangor 43000, Malaysia*
[5]*Faculty of Engineering, Universiti Teknologi Malaysia, Johor 81310, Malaysia*
[6]*Department of Computer Science, College of Computer Information Technology, American University in the Emirates, Dubai 503000, UAE*
[7]*Zayed University, Dubai, UAE*

Correspondence should be addressed to Salama A. Mostafa; salama@uthm.edu.my

Received 11 February 2021; Revised 29 March 2021; Accepted 13 April 2021; Published 23 April 2021

Academic Editor: Chalee Vorakulpipat

Currently, online organizational resources and assets are potential targets of several types of attack, the most common being flooding attacks. We consider the Distributed Denial of Service (DDoS) as the most dangerous type of flooding attack that could target those resources. The DDoS attack consumes network available resources such as bandwidth, processing power, and memory, thereby limiting or withholding accessibility to users. The Flash Crowd (FC) is quite similar to the DDoS attack whereby many legitimate users concurrently access a particular service, the number of which results in the denial of service. Researchers have proposed many different models to eliminate the risk of DDoS attacks, but only few efforts have been made to differentiate it from FC flooding as FC flooding also causes the denial of service and usually misleads the detection of the DDoS attacks. In this paper, an adaptive agent-based model, known as an Adaptive Protection of Flooding Attacks (APFA) model, is proposed to protect the Network Application Layer (NAL) against DDoS flooding attacks and FC flooding traffics. The APFA model, with the aid of an adaptive analyst agent, distinguishes between DDoS and FC abnormal traffics. It then separates DDoS botnet from Demons and Zombies to apply suitable attack handling methodology. There are three parameters on which the agent relies, normal traffic intensity, traffic attack behavior, and IP address history log, to decide on the operation of two traffic filters. We test and evaluate the APFA model via a simulation system using CIDDS as a standard dataset. The model successfully adapts to the simulated attack scenarios' changes and determines 303,024 request conditions for the tested 135,583 IP addresses. It achieves an accuracy of 0.9964, a precision of 0.9962, and a sensitivity of 0.9996, and outperforms three tested similar models. In addition, the APFA model contributes to identifying and handling the actual trigger of DDoS attack and differentiates it from FC flooding, which is rarely implemented in one model.

1. Introduction

A Distributed Denial of Service attack (DDoS) is the most common type of flooding attack, which floods computer networks. Complex network environments consist of a variety of servers, including web, Internet of Things (IoT), cloud, fog, etc., that are exposed to huge requests that slow down networks and interrupt services [1]. These attacks occur for different reasons such as financial, personal, political, ransom, and cyberwar at different security levels and

cause various attack impacts [2]. Accordingly, DDoS attacks affected nearly 2,500 organizations with 75,000 computer systems and over 100 countries with four million computers in 2010 and 2011 [3]. In the first quarter of 2016, a 602 Gbps DDoS attack was launched against the BBC website and crashed the website for several hours [4]. Basically, before the attack, the attackers (known as Demons) hack personal computer users who access the web and take over these computers. Subsequently, attackers exploit these computers by planting harmful codes or other strategies to gain control of the computers [5]. The number of these hacked computers (known as Zombies) can reach into thousands. Such a number of Zombies' creates a "botnet," which is a network of private computers that has been planted with malicious software and manipulated as a group without the owners' knowledge, e.g., to send spam. The severity of attacks depends on the size and scale of a botnet. A bigger botnet is usually associated with increasingly severe and catastrophic attacks.

There are two main types of DDoS attacks. The first type targets the Network Application Layer (NAL) such as HTTP flood, DNS flood, and FTP [6, 7]. In this type, the attacker issues vindictive or noxious bundles/packets aimed at the unfortunate casualty to cause disarray concerning the convention or any application that keeps running on it (e.g., vulnerability or defencelessness attack) [5]. The second type targets the Network and Transport layers such as UDP flood, TCP flood, and ICMP flood [8]. In all of these attacks, the attacker targets to (i) exhaust system assets, transfer speed, or the handling limit of switching to upset the network of an authentic client and (ii) exhaust the servers' assets such as memory, CPU, I/O, transmission capacity, and HDD/database transmission capacity to interfere with the administrations of legitimate clients. This study focuses on attacks targeting the Hypertext Transfer Protocol (HTTP) of the NAL.

A kind of abnormal network traffic is the Flash Crowd (FC) that causes a refusal of administration for an Internet administration's real clients [9]. The FC closely resembles the DDoS attack, whereby enormous legitimate clients simultaneously access a specific processing asset (e.g., a website). For instance, important news created worldwide, the distribution of the Olympic timetable, or organizations like Apple, Sony, and Samsung initiating a novel item brings about an unexpected flood in authentic traffic [10]. These outcomes of the ill-timed and undue conveyance of reactions by the web administration require prompt action. As DDoS attacks and FC traffic contrast in only a couple of metrics, distinguishing them is a major hurdle [11]. Researchers have suggested and actualized various cybersecurity models to defend network systems and applications from DDoS and FC attacks. However, the harmful streams disguised in authentic traffic are a scourge for these security prototypes. Many of these models cannot distinguish between real and pernicious streams with respect to negatives generations and false-positives.

An agent is a programming component or an integration of programming and equipment entities that can be executed in parallel in its clients' interest. It includes numerous helpful functions, such as learning capability, cooperation, responsiveness, and effectiveness [3]. The agent is deployed in this area either in the attacker or defense teams [12, 13]. For instance, in Kotenko et al. [14], an agent or agents are employed with an assailant system to produce and control a vast number of deceitful DDoS botnet traffic. The agent is used to oversee or handle versatile decision-making forms in the protection against DDoS attacks. The enormous hurdle in creating and strengthening the defense components of DDoS is to distinguish between the DDoS attack and an FC, in which a real action may oftentimes show up as malevolent. Cybersecurity research that focuses on distinguishing between DDoS and FC attacks has progressed over the years. Various artificial intelligence methods, such as fuzzy logic, genetic algorithm, K-Nearest Neighbor (K-NN) calculation, Bayesian networks, neural networks, software agent technology, and Support Vector Machines (SVM), are discussed in the literature.

We are inspired to design an agent-based defense model that has the ability to protect against DDoS and FC targeting the NAL. We consolidate the agent with the protective or defensive archetype. We assume that it is important to develop an effective method that detects DDoS attacks and expunge malicious traffics at the application layer level before they cause harm to the web servers and applications. We propose an Adaptive Protection of Flooding Attacks (APFA) model to protect the NAL against DDoS and FC. Four modules form the APFA model: (i) Abnormal Traffic Detection Module (ATDM), (ii) DDoS Attack Detection Module (DADM), (iii) Adaptive Traffic Control Module (ATCM), and (iv) Kalman and Bloom Filters Module (KBFM). The ATCM represents our main contribution, which integrates an adaptive agent with the belief-desire-intention (BDI) architecture to identify, classify, and control traffics of network systems. The test results of the APFA model show that the adaptive agent does not just give an upper hand by enhancing procedure value or capacity but coordinates the process of the innovative modules and improves the overall performance of the simulated network system.

We organize this paper into six sections having the first section as an introduction to the research work. In Section 2, we review the related work. Section 3 presents the research methods and materials. Section 4 illustrates the main components of the APFA model. Section 4 describes the simulation environment and testing platforms. In Section 6, we discuss the results and review the contributions and limitations of this work. Finally, Section 7 presents the conclusion and highlights a key point for future work.

2. Related Work

Several well-established studies have focused on the defense against DDoS attacks and control FC traffics that targeted the NAL. In this section, we review the details of the most effective and related well-established works that have been presented and discussed in the literature.

Shiaeles et al. [15] accomplished a DDoS attack recognition with enhanced time constraints using a

nonasymptotic fuzzy evaluator. The evaluator is implemented on average packet inter-arrival durations. The complication is divided into two units: recognition of the actual DDoS attack and identification of the IP addresses of the victims. The former task is accomplished by employing stringent, real-time boundaries for DDoS attack discovery. The latter goal is achieved using comparatively lenient constraints, which identify the IP addresses of the victims promptly, thereby starting embedded anti-attack functions on the affected hosts employing the arriving time of the packet as the primary statistic of DDoS attack detection.

Kaur et al. [16] use the "survival of the fittest" principle in which when many clients try to get scarce assets; the stronger clients overcome the weaker ones. Consequently, to replace clients with low fitness, a chain of repetitions or successive approximations is implemented using a fitness or suitability function. In this instance, GAs could be used with information captured from inbound streams of packets and in selecting optimum metrics to detect and distinguish attacks from normal packets. Katkar et al. [17] recommend using a network intrusion detection system model that uses signatures to identify DDoS attacks on HTTP servers by using shared handling and a naive Bayesian classifier. They use observational outcomes to validate the efficiency of the model. The naive Bayes classifies attacks that are slow and have 97.82% precision, and regular behavior is detected with a precision of 96.46%.

Barrionuevo et al. [18] propose an approach and an analysis of its practicability on three known attacks of service denial: Fraggle, Land, and Smurf. They solve the execution problem using the HPC techniques in the GPU to quicken the procedure and produce the outcomes. They evaluate the approach via several indices. The proposed approach achieves 40% to 70% accuracy and 60% to 83% sensitivity. The F-measure, which is employed to estimate the framework's execution, is 0.5 to 0.83. Sreeram and Vuppala [19] propose a Bio-Inspired Anomaly-based application layer DDoS attack (App-DDOS Attack) to defend against DDoS attack by using the CIDDS dataset. Furthermore, the proposed model aims to achieve fast and early detection. As shown in the results, the proposed model achieves an excellent result in defending against DDoS attacks with 99.64% accuracy. However, the proposed model lacks the ability to deal with the legitimate traffics that stream with pernicious DDoS traffics, but it has the ability to detect only limited types of flooding attacks.

A multilevel DDoS mitigation framework (MLDMF) is recommended for all levels of the IoT systems architecture [20] that is built upon the edge-, fog-, and cloud-computing levels. IoT gateways are utilized at the edge-computing level to manage and secure IoT nodes based on the SDN. An IoT management control unit (IMCU) is employed at the fog-computing level, which consists of SDN controllers and software to detect and neutralize DDoS attacks. On the other hand, the cloud-computing level analyzes the network traffics using big data and AI to protect against DDoS attacks by establishing an intelligent attack identification and mitigation structure. The simulation outcomes of the three computing level architecture of the IoT show that the edge-computing level's quick response capability, fog-computing level's state recognition feature, cloud-computing level's computing capability, and SDN's network programmability could solve the DDoS problem in IoT.

Verma and Ranga [21] present the measurable examination of the marked stream-dependent CIDDS dataset utilizing K-NN grouping and K-Means bundling calculation. Some noticeable assessment parameters are utilized to assess IDS, including accuracy, recognition rate, and false-positive rate. In another work of Verma and Ranga [22], they lead an itemized investigation of the CIDDS dataset and report the discoveries. They utilize a wide scope of familiar AI procedures to examine the multifaceted nature of the dataset. The assessment measurements that they use include recognition rate, precision, false-positive rate, kappa insights, and root mean squared deviation to appraise implemented AI approaches.

Mohamed et al. [23] come up with an identification framework of HTTP DDoS attacks in a Cloud domain that depends on Information-Theoretic Entropy and Random Forest collection learning calculation. They utilize a time-sensitive sliding window calculation to appraise the measure of randomness of the network header attributes of the approaching system traffic. At the point when the evaluated entropy surpasses its typical range, the preprocessing and the characterization exercises are activated. To evaluate the suggested methodology, they carry out different tests on the CIDDS-001 open dataset. The recommended methodology accomplishes acceptable outcomes with a precision of 99.54% and FPR of 0.46%. Moreover, the framework has been proposed to protect the cloud environment against DDoS attacks. However, the proposed framework is inefficient in handling FC, and it can only detect limited types of flooding attacks.

An agent-based methodology and programming condition (which is based on the OMNeT++ INET framework) is designed by Kotenko et al. [24] to model shared protection techniques for installation on the web to neutralize network attacks. This method is characterized by various agent groups that collaborate to neutralize malicious traffics and as a protection mechanism against attacks. Similarly, Juneja et al. [25] suggest a multi-agent architecture to identify, protect, and track the origin of a DDoS attack. While this approach is able to locate the source of a DDoS attack, a number of agents are needed to produce the best results.

Kesavamoorthy and Soundar [26] develop a technique, which uses a self-contained multi-agent system for detecting and protecting against DDoS attacks. In this technique, agents use particle swarm enhancement/optimization to attain an excellent correspondence or interaction. DDoS attacks are recognized when many connected agents are deployed to communicate new attacks to the coordinator agent. The cloud-based system protects against many types of DDoS attacks with an accuracy of 98%. A multi-agent-based distribution system identifies and prevents DDoS attacks within the ISP boundaries and is presented in the work of Singh et al. [27]. The agents and their coordinating partners implement the task of preventing the attacks in all ISPs. These agents work together by checking the incoming traffics on the edge router and using an entropy threshold-

based technique to detect the existence of DDoS attacks. If an attack occurs, the coordinator agent communicates this information with the neighboring ISPs to create a distributed protection environment. The authors adopt certain metrics to assess the performance of the defense system. However, the system's efficiency is evaluated against the system's performance in the absence of suitable metrics.

Lin et al. [28] suggest two versatile sampling calculations to gather security-associated information using agent technology. The agent has adaptive mechanisms to enhance acquisition productivity, guarantee to gather precision, and reduce the measure of gathered information. The aim of these mechanisms is to limit the impact of information capturing on the regular activities of a network. The outcomes demonstrate the benefits of the versatile security-associated information gatherer with respect to the productivity and flexibility of adaptive agents.

Generally, we can ascribe a DDoS attack as a scalable network security issue. While researchers have developed many detection and defensive mechanisms against DDoS attacks, success has been limited in implementing the mechanisms across a range of computing networks. The use of the artificial intelligence approach is limited to identifying whether clients' requests are valid or malicious based on the requests' attributes. However, the above discussions clearly enlighten the software agent's suitability as a technology that could be used in our proposed model to make the system more flexible and adaptable in dealing with the various cases of DDoS and FC targeting network traffics.

3. Materials and Methods

This section discusses the research materials and methods of this work, starting with a review of the adaptive agent architecture and mechanisms related to this work, followed by a description of the CIDDS testing dataset and its attributes. Subsequently, we explain the threats model design and the evaluation methods.

3.1. Adaptive Agent. An agent is a mix of equipment or programming elements that is responsive, for the benefit of its clients, in an autonomous manner. It has numerous helpful attributes like adaptivity, autonomy, connectivity, learning, reactivity, and proactivity. An adaptive agent provides applicability in vast domains, for example, portable processing, data recovery and processing, smart communication, media communications, and electronic commerce [29]. These agents interact in a multi-agent framework and are directed in different manners to serve particular clients or perform specific tasks. The qualities that spurred the utilization of the agent technology in this work include its self-governance, adaptation to failures, dynamic setup, autonomous decisions, situatedness, and scalability [25]. The agent may now and again endeavor to adjust to be more adept to its new or dynamic condition or to manage new or evolving objectives [30]. Contemplations of agent alteration or acclimatization incorporate what calculation can be utilized to alter the agent behavior? What is the utmost measure of progress anticipated in the agent framework? How is the framework going to stop development from going beyond control? And how to recognize and manage an alteration whose impact is not ideal? Versatile identification is the learning capacity to recognize any alteration in chance markings or configurations in an environment or system to be more adept to its condition [31]. Figure 1 demonstrates the deployment of adaptive mechanisms in agents based on the agent's dynamics and the related system.

The motivation behind the adaptation behavior can be a response to changes, evaluation of situations, or dealing with uncertainty. Adaptive procedures can be time-differing when receptive or responsive to a disturbance with a continuous interior shift of the choice procedure through repeated choice, successive choice, or audited rules [29, 32]. The reactive or responsive adaptive type is considered the most effective in this domain because it portrays the limit of the protection prototype (e.g., time) to respond against the DDoS attack. For instance, Cheng et al. [33] propose a DDoS attack recognition model that utilizes responsive adaptation in an agent to recognize and control attack streams. The agent utilizes the responsive adaptation to screen the conduct of approaching streams of information and afterward control the traffic movement.

3.2. The Testing Dataset and Parameters. Coburg Intrusion Detection Data Sets (CIDDS) is a marked stream-dependent dataset [6, 34]. It is created essentially for the assessment of IDS and IPS. The dataset comprises OpenStack and External Servers traffics. We ignore Attack ID and Attack Description's features in this study because they just offer extra insights into the executed attacks without significantly contributing to the analysis. We collect about 153,026 occurrences from the outer servers and 172,839 occurrences from the OpenStack Server information for examination. The dataset classes' occurrences are labeled or marked as expected, assailant, unfortunate casualty, suspicious, and obscure classes. Table 1 gives a representation of CIDDS dataset features.

Basically, the CIDDS dataset is chosen because it is the most recent dataset, produced in 2017; available online for free; and can simulate real-time processing due to its duration attribute. It also has the attributes of both DDoS attack and FC flooding traffics and the other existing datasets such as KDD, DARPA, and CAIDA, which lack the above attributes. Many methods have been used in defending against DDoS and FC. Each one of them used specific parameters that are suitable for the simulated systems. Table 2 presents the used parameters in building the simulated study of this work.

3.3. The Threats Model Design. This study is mostly involved with three sorts of flooding attacks or attacks, in which each is more clandestine than the previous one. (i) The assailants put forth countless HTTP solicitations to expand the framework asset and make the framework useless for the legitimate-client, which we refer to as the DDoS targeting application layer [2]. (ii) The assailants assume responsibility

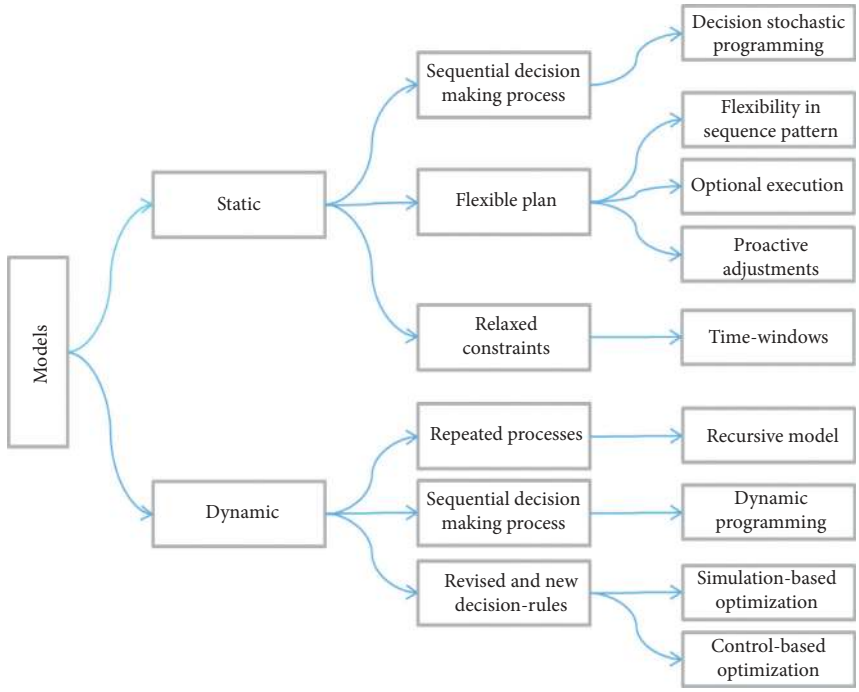

Figure 1: The models of adaptive agent.

Table 1: The CIDDS dataset attributes [34].

No.	Feature name	Feature description
1	Src IP	IP address of the source node
2	Src port	Port of the source node
3	Dest IP	IP address of the destination node
4	Dest port	Port of the destination node
5	Proto	Protocol
6	Data are first seen	Start time flow is first seen
7	Duration	Flow period
8	Bytes	Conveyed bytes
9	Packets	Conveyed packets
10	Flags	TCP flags
11	Attack description	Additional information about the attack
12	Attack type	Type of attack
13	Attack ID	Unique attack ID
14	Class	Category or label of the instance

Table 2: The testing parameters.

No.	Abbreviation	Parameters	Value
1	Window size	The size of dataset segmentation	7
2	Period	The duration of the dataset	7
3	SSM	Special sequence matrix	130
4	MCP	Model-checking period	24
5	MST	Model similarity threshold	Dynamic
6	P0	Normal traffic intensity	—
7	P1	Current traffic intensity	—
8	P2	Traffic behavior	—
9	P3	IP history log	—

for some PC machines through the web, leaving these PCs in a defenseless and helpless situation [5]. The assailants at that point begin misusing the shortcomings of these PCs by planting noxious codes or other hacking procedures to deal with the machines; they are called "Zombies." It is very simple to accomplish and certainly difficult to detect because the irregular traffic is utilized into a gathering of targets and behaves increasingly like an authentic visiting. (iii) A kind of system traffic is FC that could initiate a stop of administration for an Internet administration's legitimate-clients. The FC is closely similar to the DDoS attack, in which a specific figure of traffic requests legitimate service. For example, a site is accessed by a huge number of legitimate-clients at the same time. Breaking news produced far and wide, for example, the distribution of the Olympic calendar or organizations like Apple, Samsung, and so on, launching another product brings about an unexpected flood in a legitimate-increase in legitimate-traffics [11]. All those types of DDoS attacks are generated from the CIDDS dataset because it has the required attributes and attack scenarios.

3.4. Evaluation Metrics. In this analytical study, our system's performance is evaluated using eminent metrics, such as accuracy, precision, and sensitivity. Those measurements are assessed from the components of the confusion matrix. True-Positive (TP), True-Negative (TN), False-Positive (FP), and False-Negative (FN) are the components of a confusion matrix, where TN is the number of actual nonoccurrences of an attack. TP is the number of actual occurrences of an attack. FP is the number of inaccurately identified attack occurrences. Thus, FN is the number of inaccurately identified nonoccurrences as attack cases. Accuracy or exactness is characterized as the proportion of all effectively delegated occurrences (TP, TN) to every one of the cases (TP, TN, FP, and FN). Precision or preciseness (positive predictive quality) is the proportion of TP to a sum of TP and FP.

Sensitivity is the proportion of TP to a sum of TP and FN. Accuracy is calculated using

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}. \quad (1)$$

While precision is calculated using

$$\text{Precision} = \frac{TP}{TP + FP}. \quad (2)$$

and, sensitivity is calculated using

$$\text{Sensitivity} = \frac{TP}{TP + FN}. \quad (3)$$

4. The Adaptive Protection of Flooding Attacks Model

This work proposes the Adaptive Protection of Flooding Attacks (APFA) model, an engineered or structural expansion to protect web applications and servers against DDoS and FC attacks. It is targeted at huge-scale online organizations, including nonbusiness entryway websites. The APFA consists of four accompanying units or modules: Abnormal Traffic Detection Module (ATDM), DDoS Attack Detection Module (DADM), Adaptive Traffic Control Module (ATCM), and the Kalman Bloom Filters Module (KBFM), as shown in Figure 2. The role of each module is described in the following subsections. The base work of the model is the AL-DDoS model, which is taken from [35, 36]. Therefore, some of the model's basic parts are not detailed in this paper.

4.1. The Abnormal Traffic Detection Module. The Abnormal Traffic Detection Module (ATDM) is the first part of the APFA model. This module's major aim is to monitor and analyze the traffic to detect sudden changes in HTTP GET requests. It does not take any action if no anomalies are detected in the traffic. If it detects abnormal traffic from the incoming HTTP traffic, an "attention" signal is sent to the next module, which is the DADM, for further analyses, as shown in Figure 2. Several steps are taken before sending an attention signal starting with the measurement of the incoming traffic. This can be done in many different ways, but the APFA model measures traffic intensity by using an Auto Regression (AR) mechanism [35]. In regression, previous values affect future values. Therefore, the AR mechanism uses previously observed traffic to predict the change of traffic intensity in the future. Initially, the HTTP GET traffic stream is monitored. A time-series $\{y_1, y_2, \ldots, y_t\}$ is formed by the traffic intensity, which is studied in constant time intervals. The traffic intensity is calculated by the total number of packages received in a time interval [36]. If major changes are detected, it can potentially be a DDoS attack. The AR predicts the current traffic intensity by using

$$y_t = \sum_{k=1}^{p} a_t^k x_{t-k} + e_t. \quad (4)$$

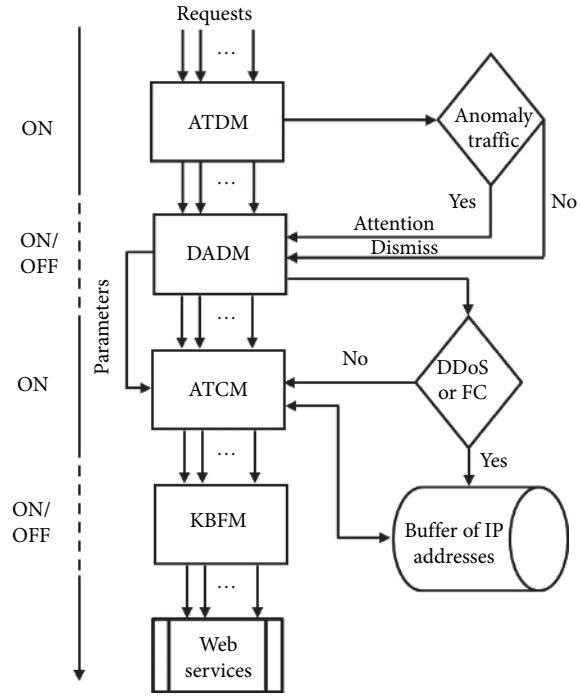

Figure 2: The APFA model.

The variable y_t predicts x_t, which is the observed value at time t. The variable a_t^k is a "constant model parameter," which means that it remains constant with time, and e_t is the observed error [36]. Secondly, at a certain time t, the difference between the observed x_t and the predicted y_t gives the residual error x_t [35].

$$d_t = |y_t - x_t|. \quad (5)$$

From the residual error at time t, a standard deviation, σ_d^2, is calculated:

$$\sigma_d^2 = \frac{\left(\sum_{i=t-p}^{t} \left(d_t - avg\left(d_{t-p}^t\right)\right)^2\right)}{p} \quad (6)$$

Subsequently, a threshold is calculated as in equation (7), which determines abnormal traffic. If d_t is greater than $k\sigma_d^2$, then, abnormal traffic is detected, and an attention signal is sent to the DDoS attack detection module. Otherwise, no abnormal traffic is detected, and the ATDM sends a "dismiss" signal to DADM, which inactivates itself, as shown in Figure 2. The constant k adjusts the sensitivity of the threshold and is set to a specific value.

$$d_t > k\sigma_d^2. \quad (7)$$

4.2. The DDOS Attack Detection Module. The DDOS Attack Detection Module (DADM) is the second part of the APFA model. It uses a trading strategy for dependably deciding a packet's source on the web. This strategy is well-known in many DDoS protection models to distinguish the legitimate origins of attacking packets existing in a network server [37]. It

contains an adaptable stream-dependent labeling plan that uses the attendant switch's load to alter stamps or labels [24]. Based on this strategy, the DADM uses Special Sequence Matrix (SSM), which denotes zero as a normal request and one as an anomaly request to give notable attributes for origin tracing the IP bundles to furnish better tracing ability [19]. Appraisals of embedded overload avoidance instruments enable this module to provide an appropriate trace-back outcome, notwithstanding when there is a substantial burden on the server. Aside from tracking DDoS attacking packets, DADM assists in enhancing the filtering or sifting of attacking traffic.

At the point when attention signals are sent from the ATDM, the DADM starts tracing the source of each IP address that sends the anomaly traffic. It then measures the mean occurrences of the associated Real-time Frequency Vector (RFV) of the traffic. The RFV holds the variation range of daily traffic for a particular server. In enormous traffics, the mean occurrence of the RFVs can be seen as the likelihood of every needed website page. Indeed, it is essential to find the value of RFV for significant traffic to deliver the progress of traffic occurrences. For the traffic model, M_1, we register RFVs possibilities:

$$p(v_i) = \frac{\sum_{i=1}^{|V|} S_{ij}}{\sum_{i=1}^{|V|} S_{ij} \sum_{j=1}^{|V|} S_{ij}}. \tag{8}$$

For a subsequent traffic model, M_2, we can determine their support values using equation (9),

$$P\left(v_i \longrightarrow v_j | v_i\right) = \frac{S_{ij}}{\sum_{i=1}^{|V|} S_{ij}}. \tag{9}$$

The certainty of the M_1 according to M_2 is obtained using equation (10). This indicates the likelihood of the upcoming traffic models, $M_1, M_2, \ldots$ from $v_i \longrightarrow v_j$:

$$P\left(v_i \longrightarrow v_j\right) = \frac{S_{ij}}{\sum_{i=1}^{|V|} S_{ij} \sum_{j=1}^{|V|} S_{ij}}. \tag{10}$$

The DADM contrasts the present prototype and the prototypes of typical traffic in the traffic model set if the present model's likelihood is more than an assumed threshold. This unusual traffic is seen as a DDoS attack model, or if the likelihood of the present prototype is lesser than an assumed threshold, this irregular traffic is viewed as a normal model [19]. In the training phase, the agent sets some of its beliefs with thresholds. These thresholds are used to reason and estimate the incoming traffic types between normal, abnormal, FC. or DDoS. as explained in Section 4.3. In addition, to distinguish the attack traffic from the typical or normal traffic for every peculiarity or anomaly traffic, the estimations of entropy on every model ($M_1, M_2, \ldots$) are determined to portray the appropriation of the approaching origins and the targeted URLs. For the purpose of the investigation, S is the RFV of source IP addresses; T is the URLs of needed website pages, numeral one as the "HTTP requests," numeral two as the "normal." According to the meanings of each DDoS attacks and FC, the entropy $En(S)$ or $En(T)$ is determined by equation (11):

$$\frac{En(S)_2}{En(T)_2} > \frac{En(S)_1}{En(S)_1} > \frac{En(S)_1}{En(S)_3} > \frac{En(S)_4}{En(S)_4}. \tag{11}$$

Hypothetically, as appearing in equation (11), normal traffic, for the most part, has the smallest proportion of entropy quality and hence, differentiates the normal traffic from the DDoS attacks. At this point, the traffic is not investigated to check for the possibility of FC.

4.3. The Adaptive Traffic Control Module. This work contributes an agent-based Adaptive Traffic Control Module (ATCM), which has a Belief-Desire-Intention (BDI) agent architecture. With the BDI architecture, the adaptive agent facilitates the task selection decisions based on mapping desires with states of beliefs. These beliefs help the agent make decisions on the course of actions required to complete the tasks. The tasks involve monitoring the behavior of the incoming traffics data and controlling the flow of the traffic. The ATCM agent has a reactive component with which it adapts the traffic through implementing three functions: anomaly traffic identification *ati* function, anomaly traffic diagnosis *atd* function, and anomaly traffic handling *ath* function. These functions process according to the values of preexisting parameters or beliefs, including traffic attack behavior, normal traffic intensity, and history log of IP address. The belief constituents include information about traffics in the normal case as well as in the abnormal case. Desires, also referred to as goals, are reflective of what the agents intend to achieve. The agent can create desires or goals explicitly or generate them during runtime. However, in the ATCM agent, the desires are predetermined by the corresponding tasks, which are explained in the following paragraphs. Lastly, intentions are interwoven with plans, which are sequences of actions structured toward achieving the goals if there is a means of achieving them. The BDI architecture of the ATCM agent reasoning cycle is as follows:

Step 1: observe the network traffic conditions and update beliefs

Step 2: deliberate some defense desires to pursue based on the updated beliefs

(i) Determine the available defense alternative desires
(ii) Filter out unrelated or unachievable desires

Step 3: generate intentions of carrying out tasks to satisfy the selected desires

Step 4: execute actions to complete the corresponding task

In addition, with these components of BDI, the agents goals are differentiated from plans. There may be several plans prepared for achieving a goal so that if one plan fails, the agent considers other plans according to the reasoning cycle. In a case wherein there are multiple plans to achieve the goal, there is a cost-based selection function so that a less time-consuming plan is selected. Figure 3 shows the architecture of the proposed ATCM and the related adaptive functions.

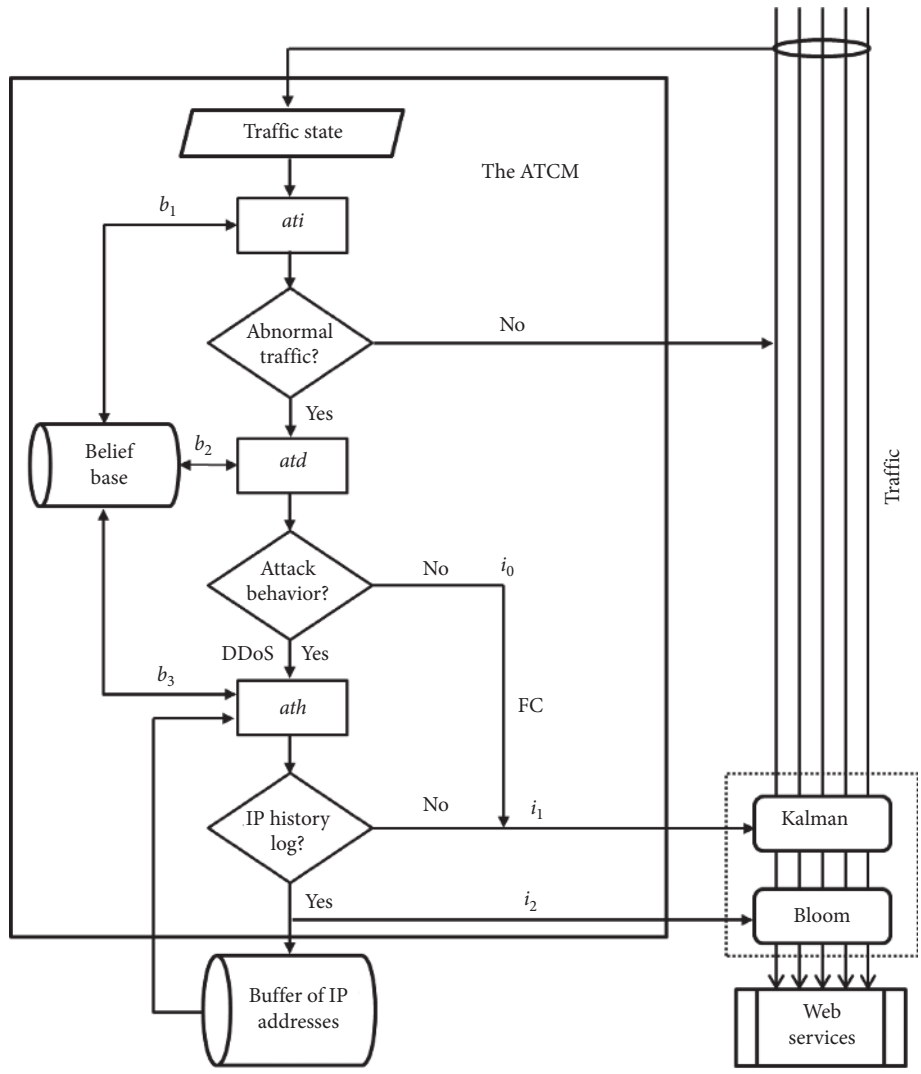

Figure 3: The architecture of the ATCM Agent.

Let the beliefs set, B, represent the network traffic parameters, which are: normal intensity, b_0; traffic intensity, b_1; traffic behavior, b_2; and IP history log, b_3. The agent's beliefs trigger the desires set, D, to react based on the traffic conditions. Three functions: *ati*, *atd*, and *ath*, filter the desires, D, and translate the D to intentions, I. The I include the options of filters, i_0; block, i_1; and lock, i_2 traffic actions. They are defined as follows:

(i) i_0 temporarily filters the traffic signals by random dropping of network requests. i_0 is invoked when FC is detected

(ii) i_1 temporarily blocks the DDoS zombie network requests. i_1 is triggered when DDoS zombie IP addresses are detected

(iii) i_2 permanently locks the DDoS demon network requests. i_2 is invoked when DDoS demon IP addresses are detected

Based on Figure 3, when the anomaly traffic with the source IP address reaches the agent of the DADM, the ATCM agent controls the incoming traffic according to the three predefined intentions. In the first step of an agent cycle, the *ati* function checks the current traffic intensity with the b_1. In case the current traffic intensity is more than b_1, it means there is an attack traffic state. In case the current traffic intensity is less than b_1, it means a normal traffic state, and the traffic is allowed to pass to the web service. Subsequently, in the second step of the agent cycle, and after it determines that the incoming traffic is a potential attack, then the second function, which is the *atd* and based on b_2 classifies the type of traffic into DDoS or FC according to equation (12).

$$r = \frac{n(\sum xy) - (\sum x)(\sum y)}{\sqrt{n\sum x^2 - (\sum x)^2}\sqrt{n\left(\sum y^2\right) - (\sum y)^2}}. \tag{12}$$

At this point, any traffic that cannot be confirmed to be DDoS is labeled as FC. In the case of FC, the agent invokes the execution of random Kalman filter, rkf function, i.e., in the KBFM to block some of the traffic in a random manner

temporarily. In the case of DDoS, the exclusive decision is sent to the agent's last function, which is *ath* for further analysis. The ath based on b_3, separates the DDoS traffic into Demons and Zombies. Then, the Demons' IP addresses are sent to the specific bloom filter for the sbf function to block the Demons permanently. Finally, the Demons' IP addresses are saved in the buffer IP address for future processing. Consequently, the Zombies' IP addresses are sent to the specific Kalman filter for the skf function to block the Zombies temporarily. All the filter functions are described in the KBFM.

4.4. Kalman and Bloom Filter Module. The Kalman and Bloom Filter Module (KBFM) comprise Kalman and Bloom filters. These filters are sequentially associated with network traffics. In the following segments, we clarify the significance and utilization of these filters.

4.4.1. Kalman Filter. The Kalman filter includes expressions that permit assessing the procedure state via productive and recursive computation such that the average of the squared error is limited [38]. In our suggested prototype, the Kalman filter is controlled by the agent. The agent sends signals to the Kalman filter for actuation or shut-off depending on the prearranged metrics and measure of approaching traffics by invoking one of the two functions. The first function is the random Kalman filter, rkf, function that performs impermanent blocks to random IP addresses. The second function is the specific Kalman filter, skf, function that performs impermanent blocks to the Zombies' IP addresses.

4.4.2. Bloom Filter. In 1970, Burton Howard Bloom created a filter named after him, called the Bloom filter, which can be described as a probabilistic data structure that is space-efficient. This filter can be utilized to test and decide whether a component is a member of a set. There is a plausibility of false-positive matches but not false-negatives. Eventually, a query can return as just "certainly not in set" or "potentially in set" in which components can be included to the set but not expelled when all things are considered as continuous events. The likelihood of false-positives becomes bigger when the number of components in the set increases [39]. In our suggested prototype, the bloom filter is controlled by the agent. It signals the bloom filter for initiation or shut-off depending on the predetermined metrics and measure of approaching traffics by invoking specific bloom filter, sbf function. This function performs permanent locking of the Demons' IP addresses.

5. Simulation Environment

This segment discusses the implementation of the simulator, the tests performed, and the execution measurements that are utilized in evaluating the APFA model. The simulator includes implementing the AL-DDoS model of Zhou et al. [36] as a base model. It also includes attack visualization and analysis modules to monitor the performance of the attack traffics and the protection models. The simulation illustrates the impact of the DDoS and FC on the application layer with and without the AL-DDoS and APFA models.

5.1. Simulator Description. We build the simulation process design based on the attributes of the CIDDS dataset, and the AL-DDoS and APFA models. We use the CIDDS dataset to generate a large number of HTTP requests, including normal and abnormal HTTP requests. We divide the dataset into four weeks and model it with predetermined settings, which we describe in the following section. We design the APFA model in an almost similar design to the AL-DDoS model, except that we add the ATCM agent and some related changes.

Figure 4 shows the complete simulator design, which starts with a connection of the dataset to the simulator and dividing the data into training and testing sets. Traffic data from the training set is fed to the ATDM to determine the simulator thresholds by monitoring and analyzing the incoming traffics during the training phase (Steps 1–4 as shown by the ellipses). These thresholds are also received by other modules and the ATCM to form the agent's initial beliefs. In the subsequent testing phase, the ATDM distinguishes between the normal and the abnormal traffic, and passes the attention or dismiss the signal to the DADM. If an attention signal is received, the DADM traces the source of IP addresses that send the anomaly traffics (Step 5). Consequently, it sends these IP addresses in the form of SSMs to the ATCM agent for further analysis. While this study contributes the ATCM agent as discussed in the previous section, the BDI architecture of the ATCM agent controls the execution of three plans (Step 6) [40].

These plans identify traffic conditions (Step 6.1), classify the traffic type (Step 6.2), and control the traffic flow (Step 6.3). These could be selected sequentially or arbitrarily based on the traffic conditions and changes in the agent's beliefs. Finally, the filtering operation that satisfies the analysis of the traffic conditions is invoked (Step 7). Figure 5 shows the sequence of the interactions between the four modules of the APFA model of the simulator.

In this diagram, the rectangles show the modules, and the squares represent the procedures of each module, whereas the arrows show the direction of processing and the interaction in a time frame as follows:

(i) User: exports the CIDDS dataset through a GUI

(ii) ATDM: monitors and analyzes the incoming traffics to set thresholds

- o sends attention signal to the DADM in the case of abnormal traffics
- o sends dismiss signal to the DADM in the case of normal traffics

(iii) DADM: traces traffic sources in the case of abnormal traffic based on the received signal

- o Attention: traces the source of abnormal traffics and saves the IP addresses

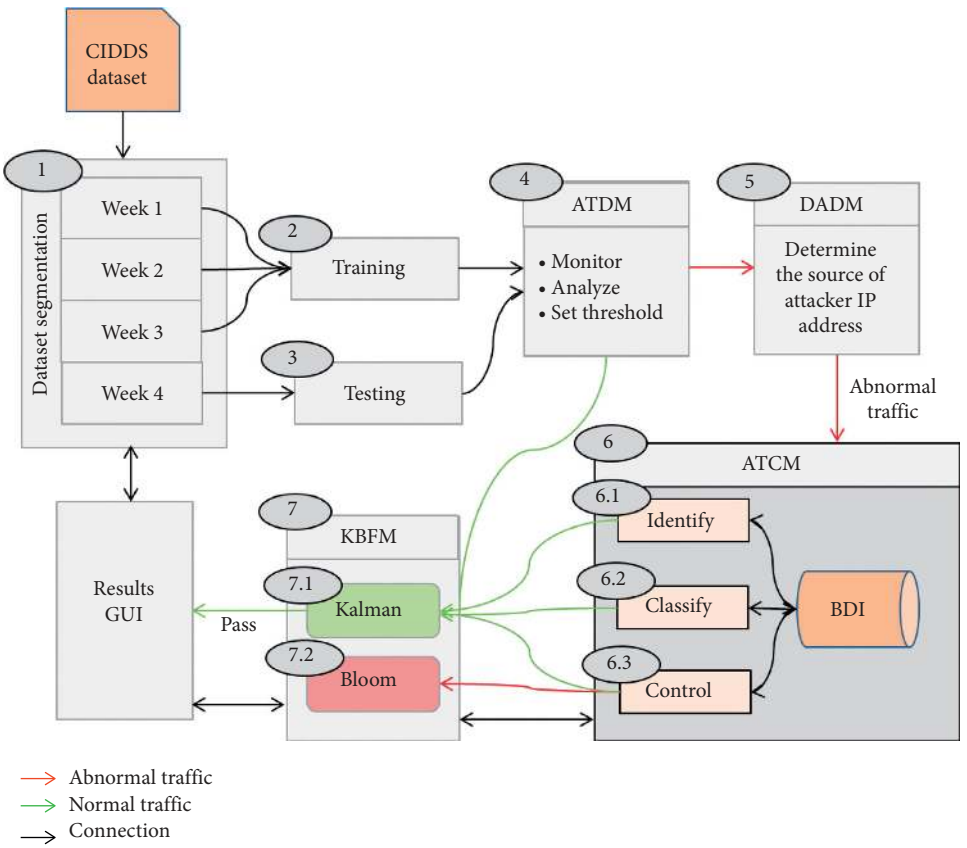

Figure 4: The steps of the simulation design.

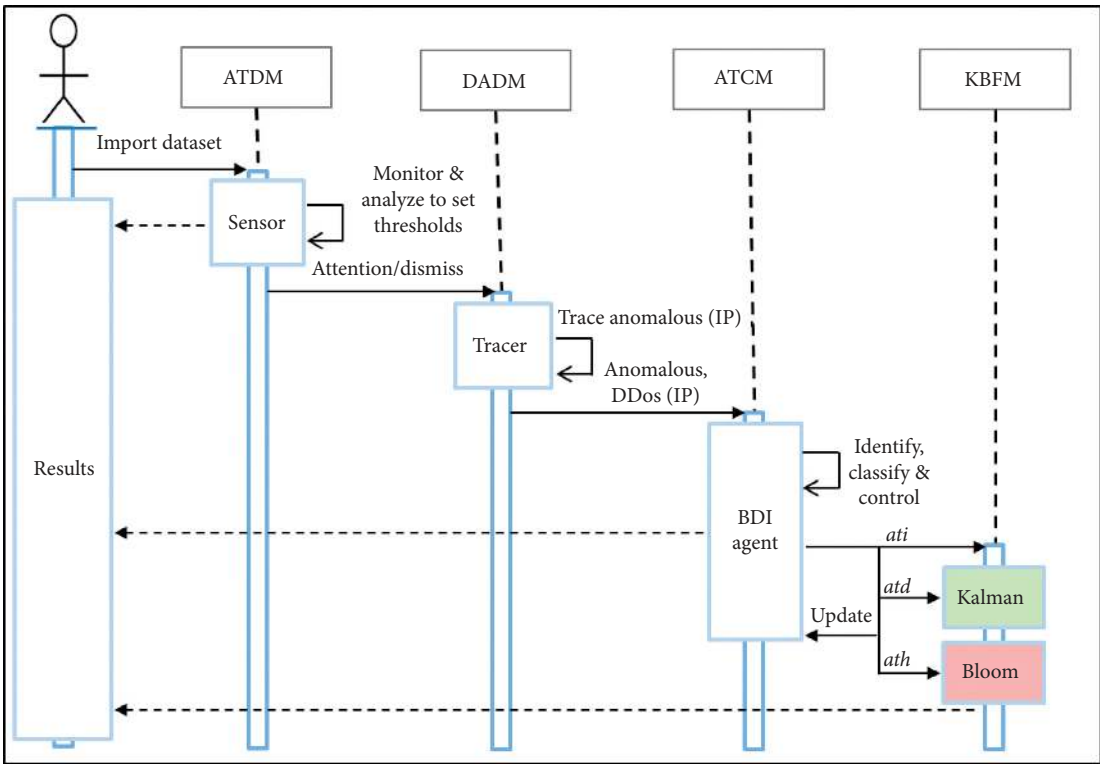

Figure 5: The sequence diagram of the APFA model simulation.

o Dismiss: stops the tracing process

(iv) ATCM: controls traffic flow in the case of abnormal traffics by invoking

o *ati*: identifies traffic conditions
o *atd*: classifies the traffic type
o *ath*: controls the traffic flow

(v) KBFM: filters traffic flow in the case of abnormal traffics by invoking

o rkf: temporarily filters the traffic according to random IP addresses and specific thresholds
o skf: temporarily filters the traffic according to specific IP addresses and specific thresholds
o sbf: permanently filters the traffic according to specific IP addresses and specific thresholds

(vi) Results: displays the information of the data analysis, processing cycles, and the simulation results through a GUI.

We specifically develop the simulator for this work by using C#, which is available on Visual Studio 2013 and Windows 7. For the implementation and testing of the simulator, the hardware used includes a 2.40 GHz Intel (*R*) Core (TM) i7-5500U processor and 16 GB RAM.

5.2. Dataset Setting. The original Coburg Intrusion Detection Data Sets (CIDDS) is a flow-based benchmark data segmented into five different groups of traffics, which are (normal, suspicious, unknown, attacker, and victim). The CIDDS dataset is used in the simulation to generate a large number of HTTP requests which include normal and abnormal HTTP requests. We neglect the Attack ID and Description features because they just give extra information about executed attacks [6]. This dataset was also used in similar recent studies, and the settings of the dataset in our work follow the work of Sreeram and Vuppala [19] and Mohamed et al. [23]. Figure 6 shows the CIDDS dataset network environment.

The performance of the IDS and IPS against flooding types of attack is specifically evaluated using the CIDDS dataset. Figure 7 shows the segmentation of the original dataset into four weeks and seven days. The week 1 folder contains 9,412 IP addresses and sends 172,838 requests; the week 2 folder consists of 8,357 IP addresses and sends 159,373 requests. The week 3 folder holds 2,605 IP addresses and sends 70,533 requests, and the week 4 folder contains 15,369 IP addresses and sends 303,024 requests. We compile these weeks in a file and reorganize the data instances accordingly. We then segment the CIDDS dataset into 60% training and 40% testing sets, as shown in Figure 7. Hence, the training and testing ratio of the CIDDS dataset is segmented according to the related work for which the comparison is made with them.

5.3. Simulation Setting. The advantages of using the CIDDS dataset in this study are that it is current and customizable. The simulation program is written with the C# programming language in a virtual environment to regenerate customized datasets that are used in this work. However, the original CIDDS dataset does not include FC labels. Subsequently, we set the ground truth of DDoS and FC traffics to train for the thresholds and methods in the simulation based on the actual data of the CIDDS dataset and statistical analysis of the data using equations (4) and (5). The analysis of the training phase results shows the average frequency of incoming requests. The high request frequency signifies the possibility of DDoS or FC traffic. Moreover, any traffic that cannot be confirmed to be DDoS and have DDoS characteristics are labeled as FC. Figure 8 shows an example of the statistical analysis, which identifies an average frequency of 37000 requests from the clock time of 4 : 20 to 19 : 20 on day 1 of week 1.

We set the simulation parameters during the training phase for both AL-DDoS and APFA models. The training set almost represents 60%, and the testing set represents the other 40% of the original dataset. It includes the Support, Confidence, and Possibility results when the window parameter is set to be 130. Correspondingly, the support, confidence, and possibility represent the values of the up triangle, diagonal, and down triangle.

We perform different tests to choose an optimal value for all the testing parameters. Figure 9 shows an example of the CIDDS dataset that generates web traffic, with original derivations and 2-step Kalman calibration. The results show the detection of noticeable deviation for the abnormal traffics.

The default M traffic model here represents the traffics of the three weeks, and it has been calculated as discussed before. Table 3 shows the support, confidence, possibility, entropy, and minimum and maximum values of the M model. The system is implemented based on these parameters, in which the period is set to 7, and the SSM is fixed to be 130.

During the training phase, the agent architecture includes three cases, anomaly traffics, DDoS traffics, and FC traffics, along with the agent's reaction setting for the three cases. The conditions of the anomaly traffic are classified based on the traffic behavior into irregular, t_0; discrete, t_2; and continuous, t_2, as shown in Table 4. This classification helps to identify the traffic types during the testing phase.

Based on Table 4 and as described in Section 4.3, the *ati* has the elementary objective of identifying whether the incoming traffic is normal or abnormal. In a scenario where 7.6428 is a threshold value for traffic intensity according to the ATDM analysis, the current incoming traffic value is updated in b_1, then it is compared with b_0. If the b_1 value is lesser than that of b_0 (i.e., 7.6428), then the case is recognized as regular traffic. If the b_1' value is higher than b_0, then the next stage of calling the *atd* is triggered to determine the traffic condition. For FC traffic, with a traffic intensity of 7.6428, we follow the same steps as the first case. Abnormal traffics are diagnosed by the *atd* according to b_2. In this scenario, based on the correlation coefficient, the traffic behaves as a discrete flow, $b_2 \longrightarrow t_1$ and atd: $b_2 \longrightarrow i_0$. As a result, the agent instructs the KBFM to invoke the rkf with 2745 capacity, temporarily filtering out 2745 IP addresses. When the volume of the DDoS traffics, which are detected in this stage, is 10838 IP addresses, each IP address sends a random number of requests. This traffic model is sent to the

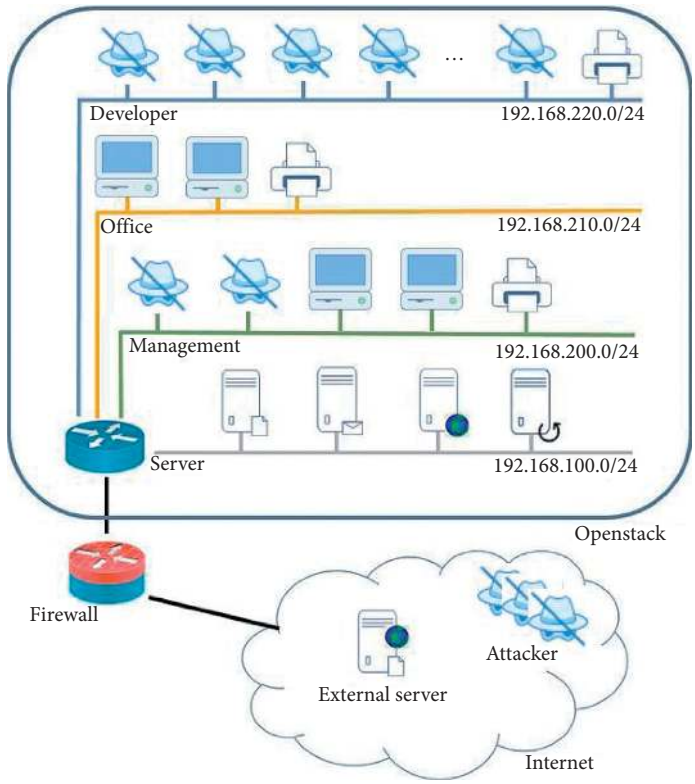

Figure 6: The CIDDS dataset network environment [6].

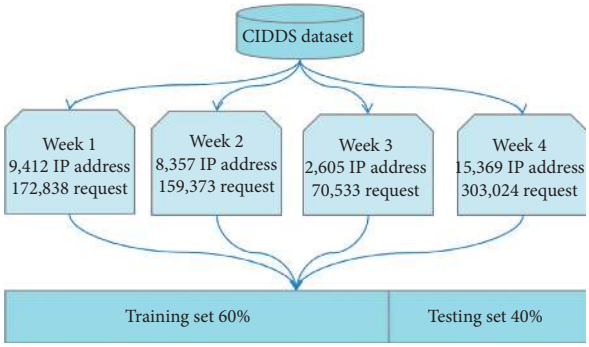

Figure 7: The sectioned parts of the CIDDS dataset.

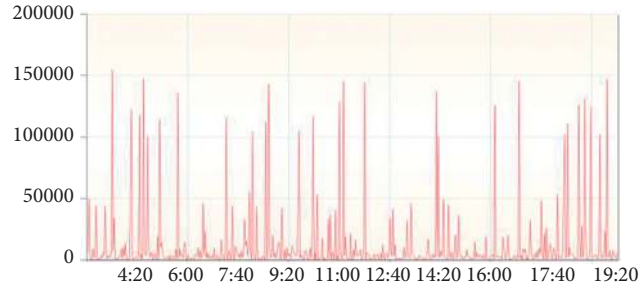

Figure 8: The average frequency of incoming requests of the training phase.

ATCM agent to identify, classify, and control the traffic according to the agent functions. When incoming traffic is 10838 and has a continuous flow, $b_2 \longrightarrow t_2$, then, the atd: $b_2 \longrightarrow i_1 \wedge i_2$ is considered as a DDoS attack. This case implies invoking ath, which handles the Zombies, r_1 and Demons, and r_2 requests, ath: $r_1 \longrightarrow i_1 \wedge r_2 \longrightarrow i_2$. The agent instructs the KBFM to invoke the skf in which ath: $r_1 \longrightarrow skf$ and the sbf in which ath: $r_2 \longrightarrow sbf$ with the corresponding SSM information, which temporarily locks the r_1 and permanently blocks the r_2.

6. Results and Discussion

The test results evaluate the performance of the APFA model. Then, the performance of the model is compared with three similar models of Sreeram and Vuppala [19], Mohamed et al. [23], and Zhou et al. [36]. We perform two tests on the CIDDS dataset to evaluate the APFA model in which the CIDDS dataset generates normal and anomaly traffics. We perform the first test for the AL-DDoS model, which only detects normal and DDoS traffics. We conduct the second test for the APFA model, which detects normal, DDoS, and FC traffics. The data of week 4 (after the modification, it becomes 40% of the dataset as explained in Section 5.2) are used for testing the model. They contain a discrete and random series of incoming requests, including DDoS and FC targeting the NAL. Table 5 shows the daily frequency of the incoming requests of week 4. The traffics of week 4 are divided into seven days, starting with traffic day 1 with 38,919 requests and ending with day 7 with 33,228 requests. We observe from the table that day 7 has visibly lower requests than the daily average incoming requests, which are 43,289, and day 4 has visibly greater requests than the daily average of incoming requests.

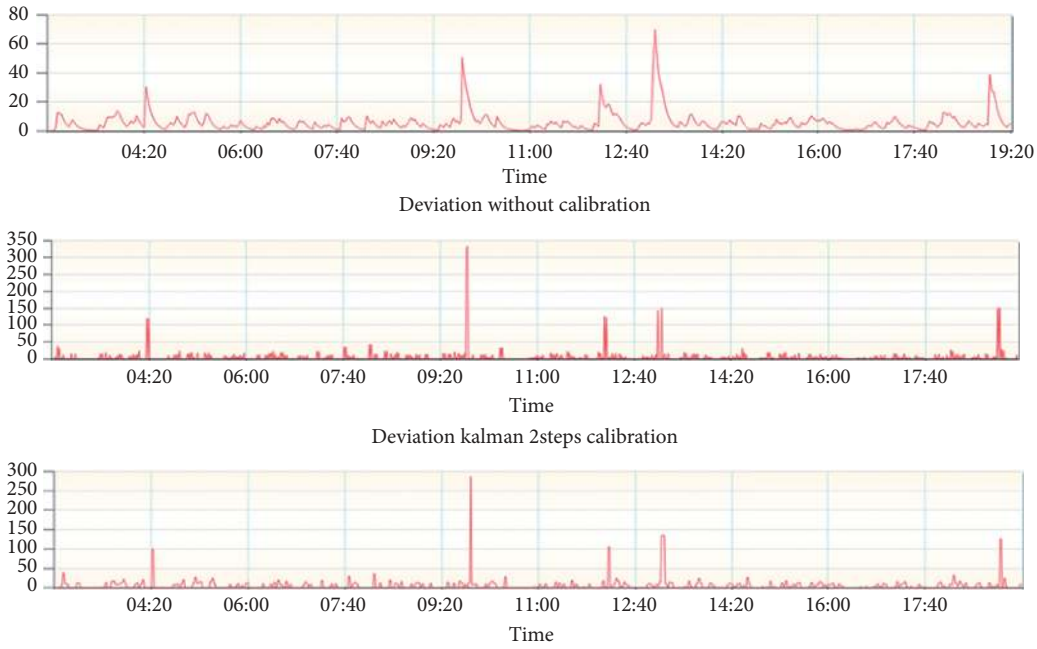

Figure 9: A sample of web traffics from original derivations and 2-step Kalman.

Table 3: Values of the M traffic model.

	Support	Confidence	Possibility	Entropy	Min	Max
M	0.0000	0.0021	0.050	7.6428	0.00000	244,557.99

Table 4: Agent parameters setting.

Traffic conditions	Traffic	Behavior
Anomaly	13,583	t_0
FC	2,745	t_2
DDoS	10,838	t_2

Table 5: The average incoming requests in the testing phase.

Day	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7
Frequency	38,919	47,328	44,921	56,991	39,835	41,802	33,228

6.1. Results of the AL-DDoS Model. The performance of the AL-DDoS base model is calculated according to the window size, period, and SSM parameters for every execution of external traffic data. The AL-DDoS model performance is evaluated based on correctly classifying traffic instances into normal, and DDoS attack traffics only. The volume of attack traffics detected by the AL-DDoS model is 26,8496 requests triggered by 13,583 IP addresses. Subsequently, the results show that the AL-DDoS model detects DDoS attacks and blocks the IP addresses with an accuracy of 99.13%, precision of 99.14%, and sensitivity of 99.99%. However, the AL-DDoS model lacks handling FC and Zombies traffics and considers all DDoS traffic as Demons.

6.2. Results of the APFA Model. The results of the APFA also present information about the number of anomaly requests of the same week 4 from the dataset. The data are first passed through the *ati* function in the identify traffic conditions phase of the ATCM agent. The *ati* detects a total number of 34,528 requests as normal and 268,496 as abnormal according to the traffic intensity parameters with a total cost of 15,369 cycles. Then, the attack requests are classified by the *atd* function, in the classify traffic type phase, into 264,551 requests as DDoS and 3,945 requests as FC, with a total cost of 13,583 cycles. The *ath* function, in the control traffic flow phase, controls the traffic according to the DDoS types of 175,624 Demons and 88,927 Zombies, with a total cost of 10,838 cycles. Then, the KBFM implements the required blocking and locking of the requests. Table 6 shows the input, processing, and output of each function in the ATCM agent.

Figure 10 shows the classification results of the daily attack requests of week 4 by DDoS and FC. The average daily DDoS

Table 6: The results of the agent run cycle during week 4.

No.	Run phase	Cost	Input req.	Output			
				Normal		Abnormal	
1	Identify normal/abnormal	15,369	303,024	IP	Req.	IP	Req.
				1,786	34,528	13,583	268,496
				DDoS		FC	
2	Classify DDoS/FC	13,583	268,496	IP	Req.	IP	Req.
				10,838	264,551	2,745	3,945
				Demons		Zombies	
3	Control demons/zombies	10,838	264,551	IP	Req.	IP	Req.
				7,186	175,624	3,670	88,927

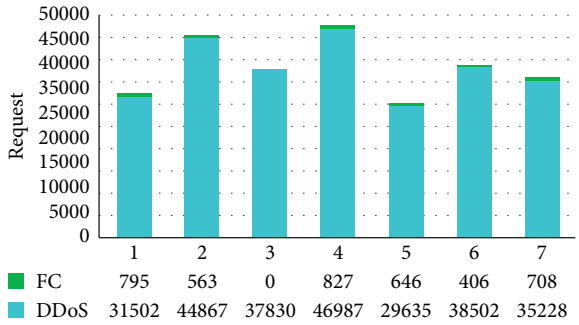

Figure 10: The daily traffics of DDoS and FC.

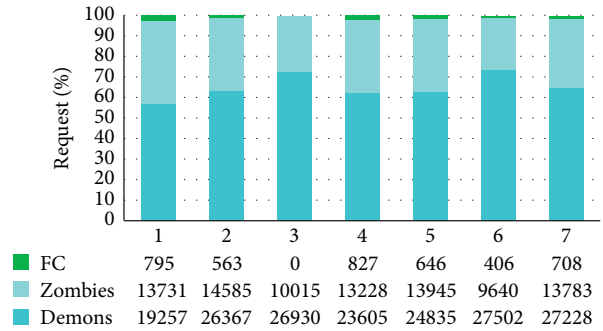

Figure 11: The daily traffics of Demons and Zombies.

requests are 37,793, which is almost 96% of the average daily abnormal requests, while the average daily FC requests is 563, which is almost 2% of the average daily abnormal requests.

Figure 11 shows the number of Demons and Zombies requests by the DDoS traffic. The average daily requests of Demons is 25,103, which is almost 66% of the DDoS requests, while the average daily requests of Zombies is 12,703, which is almost 44% of the DDoS requests.

In general, the results show that the APFA model is able to detect and distinguish the DDoS and FC traffics. It then recognizes Demon and Zombie requests of the DDoS traffic. The phase that is responsible for identifying the possibility of abnormal traffic achieves the results of 99.11% accuracy, 99.14% precision, and 99.99% sensitivity. The phase that is responsible for classifying DDoS and FC traffics achieves the results of 99.92% accuracy, 99.85% precision, and 99.96% sensitivity. The phase that is responsible for controlling Demons and Zombies traffics achieves the results of 99.91% accuracy, 99.89% precision, and 99.93% sensitivity. Ultimately, the APFA model achieves an overall accuracy of 99.64%, precision of 99.62%, and sensitivity of 99.96%. Table 7 shows the performance results of the APFA model.

Figure 12 shows the daily performance results of the APFA model. As observed from the figure, the APFA model's performance improves day by day due to the system's ability to progress its adaptive behavior with time.

6.3. Analysis and Discussion. In the deep view of the Internet network, there are many components that participate in making up the web application framework. The HTTP requests sent from web clients are processed by a web server and forwarded to the application server based on many

Table 7: The performance results of the APFA model.

Evaluation metrics			
Run phase	Accuracy %	Precision %	Sensitivity %
Identify normal/abnormal	99.11	99.14	99.99
Classify DDoS/FC	99.92	99.85	99.96
Control demons/zombies	99.91	99.89	99.93
Overall results	99.64	99.62	99.96

configuration parameters like URL path prefix. These requests are directed to one of the web applications hosted by the application server. A DDoS attack is a malicious event that targets web servers without the need for internal system access. Consequently, the attack is not easily detected in its early stage. The attack entails the involvement of a huge army of Zombies to cause conceivable damage to the network. Critical attacks include concentrating a huge number of nodes as a single target to inflict devastating damage to users and completely overwhelm the network. Another type of flooding traffics, which is FC, is depicted as network traffic that is quite similar to DDoS traffic, but it comes from valid users when a huge number of them access a particular website simultaneously.

The benchmarking works of Sreeram and Vuppala [19], Mohamed et al. [23], and Zhou et al. [36] only deal with two types of traffics, which are normal traffic and attack traffic. The AL-DDoS base model of Zhou et al. [25] only detects anomaly traffic requests, determines the source of each IP address, saves these IP addresses in a bloom filter, and locks

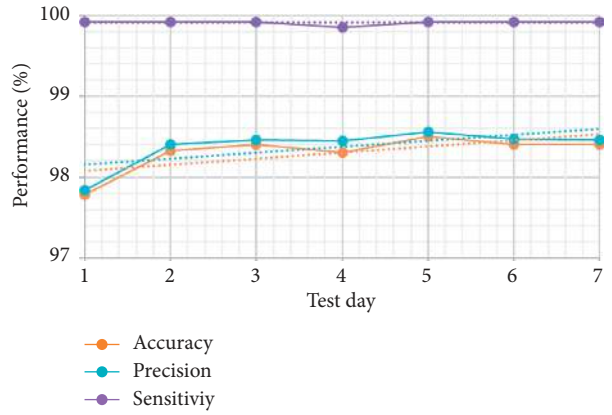

Figure 12: The results of the daily performance.

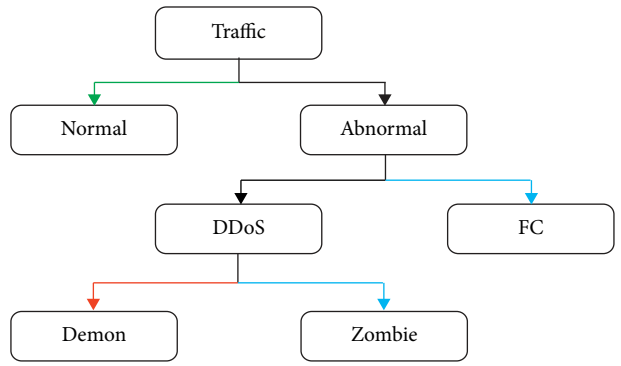

Figure 13: The hierarchical traffic control of the APFA model.

Table 8: The performance comparison with the benchmark models.

Model	Technique	Dataset	Traffic type				Accuracy %
			FC	DDoS	Zombies	Demons	
Sreeram and Vuppala [19]	Bat algorithm	CIDDS	—	✓	—	—	94.80
Mohamed et al. [23]	Random forest	CIDDS	—	✓	—	—	99.54
Zhou et al. [36]	Statistical	CIDDS	✓	✓	—	—	99.13
APFA model	Adaptive agent and statistical	CIDDS	✓	✓	✓	✓	99.64

all the anomaly traffic requests. However, among those IP addresses, there are many cases of FC and Zombies' requests that belong to legitimate users yet are confined to permanent lock. Subsequently, this work proposes an Adaptive Protection of Flooding Attacks (APFA) model that identifies abnormal traffic requests and then further classifies the abnormal traffic requests to DDoS and FC. It then further classifies the DDoS to Demons and Zombies and applies control procedures to temporarily block FC and Zombies' IP addresses and permanently lock Demons IP addresses. Figure 13 shows the hierarchy of the APFA model control to the NAL against flooding traffics.

Researchers have proposed, developed, and implemented numerous techniques to safeguard the NAL against DDoS and FC attacks. However, hidden malicious traffic behind valid traffic and the FC continue to plague these defense models. Many of these models are unable to differentiate between valid and invalid malicious traffics positively. In this paper, we proposed the APFA model, the performance of which is evaluated by comparing it with three benchmark models. The three models are tested for similar properties and in similar conditions, and there is no bias to declare. The comparison results are summarized in Table 8, which show that the APFA model outperforms the other three models.

Eventually, defense methods are continuously evolving to improve and protect networks and computer infrastructures. The APFA model, like any other model, represents another attempt to provide variable effectiveness against DDoS attacks and FC flooding. Three sources of limitations need to be highlighted according to the scope of this work. Firstly, this work does not consider the processing time in the evaluation in which the adaptation and decision-making capabilities of the agent might slow down the

performance of the model compared with the other tested models. Secondly, the model is only tested using the CIDDS dataset, which could be another constraint on the evaluation. Finally, the testing of the model does not cover the low-rate cases of DDoS attacks that are difficult to discover with existing solutions. Nevertheless, such DDoS attacks have no harmful impact on real-world network systems.

7. Conclusion and Future Work

Progressively, there are assortments of administrations and applications that utilize cyberspace, including web applications, cloud-computing applications, and Internet of things applications. DDoS attack and FC flooding could be a legitimate annoyance for the cybersecurity of network systems. The advancement of innovative communication technology of the current computer applications has brought along the danger of these sorts of threats. Subsequently, various investigations have focused on these threats to embed variable protection prototypes. The proposed mainstream models lack the ability to deal with illegitimate DDoS traffics, which are accompanied by FC traffics. They permanently lock all DDoS traffic and treat legitimate traffic of FC and Zombies as Demons. Consequently, this paper proposes an Adaptive Protection of Flooding Attacks (APFA) model, which attempts to protect the NAL against DDoS attack and FC flooding and solve the problem of permanently locking the traffics of legitimate users. The APFA model consists of the Abnormal Traffic Detection Module (ATDM) and DDoS Attack Detection Module (DADM) that are adopted from the AL-DDoS model of Zhou et al. (2014). It further includes a new Adaptive Traffic Control Module (ATCM) and Kalman and Bloom Filters Module (KBFM). Our main contribution is the ATCM module, which integrates an adaptive agent to recognize and isolate normal from abnormal traffic and hinders all ill-conceived traffics. The APFA model is implemented and tested using the CIDDS dataset to produce standard scenarios targeting web servers. The test results show that the APFA model outperforms three similar models and achieves an accuracy of 0.9964, a precision of 0.9962, and a sensitivity of 0.9996. Two points of improvement can be further investigated, which are the effect of cost function on agent adaptive behavior and enabling the DADM to detect low rate and FC-like DDoS attack patterns. In addition, putting and testing the APFA model online could furnish greater confidence in its capability to perform in real-time.

Data Availability

The used dataset for this research is available online and has a proper citation within the paper contents.

Conflicts of Interest

The authors declare that they have no conflicts of interest to be addressed related to this work.

Acknowledgments

The authors would like to thank the Center of Intelligent and Autonomous Systems (CIAS), Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia (UTHM) for supporting this work.

References

[1] N. S. Rao, K. C. Sekharaiah, and A. A. Rao, "A survey of distributed denial-of-service (DDoS) defence techniques in ISP domains," in *Innovations in Computer Science and Engineering*, pp. 221–230, Springer, Singapore, Asia, 2019.

[2] Z. K. Maseer, R. Yusof, N. Bahaman, S. A. Mostafa, and C. F. M. Foozy, "Benchmarking of machine learning for anomaly based intrusion detection systems in the CICIDS2017 dataset," *IEEE Access*, vol. 9, pp. 22351–22370, 2021.

[3] B. A. Khalaf, S. A. Mostafa, A. Mustapha, M. A. Mohammed, and W. M. Abduallah, "Comprehensive review of artificial intelligence and statistical approaches in distributed denial of service attack and defense methods," *IEEE Access*, vol. 7, pp. 51691–51713, 2019.

[4] S. Khandelwal, Cybercrime technical report, 2016, http://thehackernews.com/2016/01/biggest-ddos-attack.html.

[5] A. Zulhilmi, S. A. Mostafa, B. A. Khalaf, A. Mustapha, and S. S. Tenah, "A comparison of three machine learning algorithms in the classification of network intrusion," in *Proceedings of the International Conference on Advances in Cyber Security*, pp. 313–324, Penang, Malaysia, July 2020.

[6] M. Ring, S. Wunderlich, D. Grüdl, D. Landes, and A. Hotho, "Flow-based benchmark data sets for intrusion detection," in *Proceedings of the 16th European Conference on Cyber Warfare and Security*, pp. 361–369, Dublin, Ireland, June 2017.

[7] O. S. Babatunde, A. R. Ahmad, S. A. Mostafa et al., "A smart network intrusion detection system based on network data analyzer and support vector machine," *International Journal of Emerging Trends in Engineering Research*, vol. 8, no. 1, pp. 213–220, 2020.

[8] K. Lee, J. Kim, K. H. Kwon, Y. Han, and S. Kim, "DDoS attack detection method using cluster analysis," *Expert Systems with Applications*, vol. 34, no. 3, pp. 1659–1665, 2008.

[9] S. Behal, K. Kumar, and M. Sachdeva, "Characterizing DDoS attacks and flash events: review, research gaps and future directions," *Computer Science Review*, vol. 25, 2017.

[10] B. A. Khalaf, S. A. Mostafa, A. Mustapha, and N. Abdullah, "An adaptive model for detection and prevention of DDoS and flash crowd flooding attacks," in *Proceedings of the 2018 International Symposium on Agent, Multi-Agent Systems and Robotics (ISAMSR)*, pp. 1–6, IEEE, Putrajaya, Malaysia, August 2018.

[11] S. Yu, W. Zhou, W. Jia, S. Guo, Y. Xiang, and F. Tang, "Discriminating DDoS attacks from flash crowds using flow correlation coefficient," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 6, pp. 1073–1080, 2012.

[12] A. Bhandari, A. L. Sangal, and K. Kumar, "Characterizing flash events and distributed denial-of-service attacks: an empirical investigation," *Security and Communication Networks*, vol. 9, no. 13, pp. 2222–2239, 2016.

[13] S. Bhatia, G. Mohay, A. Tickle, and E. Ahmed, "Parametric differences between a real-world distributed denial-of-service attack and a flash event," in *Proceedings of the 2011 Sixth International Conference on Availability, Reliability and*

Security (ARES), pp. 210–217, IEEE, Vienna, Austria, August 2011.

[14] I. Kotenko and A. Ulanov, "Agent-based simulation of DDOS attacks and defense mechanisms," *International Journal of Computing*, vol. 4, pp. 113–123, 2014.

[15] S. N. Shiaeles, V. Katos, A. S. Karakos, and B. K. Papadopoulos, "Real time DDoS detection using fuzzy estimators," *Computers & Security*, vol. 31, no. 6, pp. 782–790, 2012.

[16] H. Kaur, G. Singh, and J. Minhas, "A review of machine learning based anomaly detection techniques," *International Journal of Computer Applications Technology and Research*, vol. 2, no. 2, pp. 1307–1319, 2013.

[17] V. Katkar, A. Zinjade, S. Dalvi, T. Bafna, and R. Mahajan, "Detection of DoS/DDoS attack against HTTP servers using naive Bayesian," in *Proceedings of the 15th International Conference on Computing Communication Control and Automation*, pp. 280–285, IEEE, Pune, India, February 2015.

[18] M. Barrionuevo, M. Lopresti, N. Miranda, and F. Piccoli, "An anomaly detection model in a LAN using K-NN and high-performance computing techniques," *Argentine Congress of Computer Science Journal*, vol. 790, no. 17, pp. 219–228, 2017.

[19] I. Sreeram and V. P. K. Vuppala, "HTTP flood attack detection in application layer using machine learning metrics and bio inspired bat algorithm," *Applied Computing and Informatics*, vol. 15, 2017.

[20] Q. Yan, W. Huang, X. Luo, Q. Gong, and F. R. Yu, "A multi-level DDoS mitigation framework for the industrial internet of things," *IEEE Communications Magazine*, vol. 56, no. 2, pp. 30–36, 2018.

[21] A. Verma and V. Ranga, "Statistical analysis of CIDDS-001 dataset for network intrusion detection systems using distance-based machine learning," *Procedia Computer Science*, vol. 125, pp. 709–716, 2018.

[22] A. Verma and V. Ranga, "On evaluation of network intrusion detection systems: statistical analysis of CIDDS-001 dataset using machine learning techniques," *Pertanika Journal of Science & Technology*, vol. 26, no. 3, 2018.

[23] I. Mohamed, K. Afdel, and M. Belouch, "Detection system of HTTP DDoS attacks in a cloud environment based on information theoretic entropy and random forest," *Security and Communication Networks*, vol. 2018, Article ID 1263123, 13 pages, 2018.

[24] I. Kotenko and A. Ulanov, "Agent-based simulation of distributed defense against computer network attacks," in *Proceedings of the 20th European Conference on Modelling and Simulation*, pp. 1–6, Springer, Bonn, Germany, May 2006.

[25] D. Juneja, R. Chawla, and A. Singh, "An agent-based framework to counter attack DDoS attacks," *International Journal of Wireless Networks and Communications*, vol. 1, no. 2, pp. 193–200, 2009.

[26] R. Kesavamoorthy and K. R. Soundar, "Swarm intelligence based autonomous DDoS attack detection and defense using multi agent system," *Cluster Computing Journal*, vol. 7, no. 11, pp. 1–8, 2018.

[27] K. Singh, K. Singh Dhindsa, and B. Bhushan, "Performance analysis of agent based distributed defense mechanisms against DDOS attacks," *International Journal of Computing*, vol. 17, no. 1, pp. 15–24, 2018.

[28] H. Lin, Z. Yan, and Y. Fu, "Adaptive security-related data collection with context awareness," *Journal of Network and Computer Applications*, vol. 126, no. 3, pp. 88–103, 2019.

[29] S. A. Mostafa, S. S. Gunasekaran, M. S. Ahmad, A. Ahmad, M. Annamalai, and A. Mustapha, "Defining tasks and actions complexity-levels via their deliberation intensity measures in the layered adjustable autonomy model," in *Proceedings of the 2014 International Conference on Intelligent Environments*, pp. 52–55, IEEE, Shanghai, China, June 2014.

[30] L. Xiong, S. Goryczka, and V. Sunderam, "Adaptive, secure, and scalable distributed data outsourcing: a vision paper," in *Proceedings of the 2011 Workshop on Dynamic Distributed Data-Intensive Applications, Programming Abstractions, and Systems ACM*, pp. 1–6, San Jose, CA, USA, June 2011.

[31] D. Chefrour, "Developing component based adaptive applications in mobile environments," *Applied Computing Journal*, vol. 77, no. 19, pp. 1146–1150, 2005.

[32] A. Evesti, H. Abie, and R. Savola, "Security measuring for self-adaptive security," in *Proceedings of the 2014 European Conference on Software Architecture Workshops*, pp. 5–11, IEEE, Vienna, Austria, August 2014.

[33] J. Cheng, C. Zhang, X. Tang, V. S. Sheng, Z. Dong, and J. Li, "Adaptive DDoS attack detection method based on multiple-Kernel learning," *Security and Communication Networks*, vol. 2018, Article ID 5198685, 2018.

[34] "CIDDS, coburg-intrusion-detection-data-sets," 2017, https://www.hs-coburg.de/forschung-kooperation/forschungsprojekte-oeffentlich/ingenieurwissenschaften/cidds-coburg-intrusion-detection-data-sets.html.

[35] S. Wen, W. Jia, W. Zhou, W. Zhou, and C. Xu, "CALD: surviving various application-layer DDoS attacks that mimic flash crowd," in *Proceedings of the 2010 4th International Conference onNetwork and System Security (NSS)*, pp. 247–254, IEEE, Victoria, Australia, September 2010.

[36] W. Zhou, W. Jia, S. Wen, Y. Xiang, and W. Zhou, "Detection and defense of application-layer DDoS attacks in backbone web traffic," *Future Generation Computer Systems*, vol. 38, pp. 36–46, 2014.

[37] M. De Donno, N. Dragoni, A. Giaretta, and A. Spognardi, "DDoS-capable IoT malwares: comparative analysis and mirai investigation," *Security and Communication Networks*, vol. 2018, 2018.

[38] K. Fujii, "Extended Kalman filter," *Refernce Manual*, pp. 14–22, 2013.

[39] J. Bruck, J. Gao, and A. Jiang, "Weighted bloom filter," in *Proceedings of the 2006 IEEE International Symposium onInformation Theory*, pp. 2304–2308, IEEE, Seattle, WA, USA, July 2006.

[40] S. A. Mostafa, A. Mustapha, A. A. Hazeem, S. H. Khaleefah, and M. A. Mohammed, "An agent-based inference engine for efficient and reliable automated car failure diagnosis assistance," *IEEE Access*, vol. 6, pp. 8322–8331, 2018.