# Traffic Seasonality aware Threshold Adjustment for Effective Source-side DoS Attack Detection

**Giang-Truong Nguyen[1], Van-Quyet Nguyen[1,2], Sinh-Ngoc Nguyen[1], and Kyungbaek Kim[1*]**
[1] Department of Electronics and Computer Engineering, Chonnam National University
Gwangju, South Korea
[2] Faculty of Information Technology, Hung Yen University of Technology and Education
Hung Yen, Viet Nam
[E-mail : truongnguyengiang.bk@gmail.com, quyetict@utehy.edu.vn, sinhngoc.nguyen@gmail.com, kyungbaekkim@jnu.ac.kr]
*Corresponding author: Kyungbaek Kim

*Received July 25, 2018; revised September 19, 2018; revised October 3, 2018; revised November 11, 2018; accepted December 7, 2018; published May 31, 2019*

## Abstract

In order to detect Denial of Service (DoS) attacks, victim-side detection methods are used popularly such as static threshold-based method and machine learning-based method. However, as DoS attacking methods become more sophisticated, these methods reveal some natural disadvantages such as the late detection and the difficulty of tracing back attackers. Recently, in order to mitigate these drawbacks, source-side DoS detection methods have been researched. But, the source-side DoS detection methods have limitations if the volume of attack traffic is relatively very small and it is blended into legitimate traffic. Especially, with the subtle attack traffic, DoS detection methods may suffer from high false positive, considering legitimate traffic as attack traffic. In this paper, we propose an effective source-side DoS detection method with traffic seasonality aware adaptive threshold. The threshold of detecting DoS attack is adjusted adaptively to the fluctuated legitimate traffic in order to detect subtle attack traffic. Moreover, by understanding the seasonality of legitimate traffic, the threshold can be updated more carefully even though subtle attack happens and it helps to achieve low false positive. The extensive evaluation with the real traffic logs presents that the proposed method achieves very high detection rate over 90% with low false positive rate down to 5%.

***Keywords:*** DoS attack, DoS detection, source-side detection, adaptive threshold, traffic seasonality.

## 1. Introduction

Denial of Service (DoS) attack has been considered as one of the main threats for causing serious impact on networking infrastructure. The attackers could launch this kind of attack by commanding malware infested private computers (botnets) to send a large volume of network traffic to a victim. Specifically, attackers would exploit some specific protocols, like Domain Name Server (DNS) or Network Time Protocol (NTP), to send corresponding network traffic to a victim. Being overwhelmed by these illegitimate ones, victim's bandwidth, as well as their server resource, could be exhausted, which makes legitimate users be unable to use internet services.

Because of its serious impact, to detect DoS attack, a large number of methods [1] are proposed, which could mitigate its effect. Currently, most of these proposals are situated at the victim side [2-4, 13, 14, 21, 22]. Although these methods are proved to be effective to detect DoS attack by their evaluations, they still have some natural drawbacks. The first drawback is the late detection, that is, DoS attack is only detected after it has occurred for a significant duration. As a result, useless network traffic could consume bandwidth nearby a victim before any solutions could be taken into an action. The second drawback is the difficulty to trace back locations of attackers and botnets. In a simple attacking script, attacks are launched from botnets to a victim directly, and network administrators may look up the addresses of botnets and attackers. However, in real situations, an attacking scenario becomes more complicating, and attackers can order botnets to use spoofed IP addresses, then send attack requests to a victim directly or through a medium server to amplify the volume of attack traffic. In this case, it is very difficult to trace back botnets and attackers.

To mitigate the drawback of victim-side detection, source-side detection methods are proposed [5-8, 11, 23], which situate closer to the attacking source. Unarguably, these methods could detect attack traffic much earlier than victim-side methods and they also could find out the addresses of the infected computers even they use IP spoofing or not [9]. Until now, source-side detection has been studied with various methods, like machine learning-based one [5], which detects attack by learning some statistical features and statistic based ones [8, 11], which are based on the incoming and outgoing traffic statistics. However, regarding the source side, because the volume of attack traffic is quite small, it could easily mix up with legitimate traffic [23], which causes the difficulty for statistic based methods [8, 11]. Besides, these two kinds of traffic are quite similar to each other, and it could raise problems for machine learning-based method as well [5].

Another kind of source-side detection method is threshold based detection, which is one of the most practical methods to deal with DoS attack from both source-side and victim-side [1, 4, 24]. When DoS attack happens, the attack traffic may form transient peaks during a short period of time, and whenever this peak of attack traffic exceeds the detection threshold set by network administrators, a notification of attack could be raised. Usually, on victim-side, static threshold is viable, because there is significant difference between attack traffic and legitimate traffic. However, the static threshold approach may not be suitable for detecting subtle attack peaks on source-side which can be easily mixed with the fluctuated legitimate traffic. If the static threshold is too high, the small attack peaks are not detected. On the other hands, if the static threshold is too low, legitimate traffic peaks are mis-considered as attack. That is, high false positive may occur.

Recently, an adaptive threshold method has been proposed to improve the detection rate of subtle attack peaks [16]. The adaptive threshold is updated by using both the previous threshold and the volume of the current observed traffic. However, when attack happens, the current observed traffic includes both legitimate and attack traffic and the adjustment process may be affected by attack traffic implicitly. According to this, this method suffers from high false positive, even though it achieves high detection rate. If we can separate the correct volume of legitimate traffic from the attack traffic during attack happens, we can improve the performance of source-side DoS detection with an adaptive threshold method.

In this paper, we propose an effective source-side DoS detection with a traffic seasonality aware adaptive threshold, in order to detect subtle attack peaks more properly with low false positive. The threshold is adjusted based on the observed and estimated normal traffic volume, which is obtained by the traffic seasonality learned from the periodical traffic volume statistics. Specifically, with some specific protocols obtained from the network traffic (such as DNS or NTP), we get the observed traffic, which is measured by the number of requests, for each equal period. And, during periods without any attack, we learned the seasonality of legitimate traffic. For example, considering a company environment, the employees could use the network more frequently in the daytime than in the evening. This seasonal behavior makes the network traffic volume to increase in the morning, then decreases gradually from the afternoon until evening. As a result, if the legitimate traffic volume is recorded in every equal time periods over a long duration, we could model the seasonality of traffic. This seasonality pattern could provide us the estimation of normal traffic volume under an attacking period based on the legitimate one from previous periods. By estimating the legitimate traffic carefully, the adaptive threshold is less affected by the volume of attack traffic, and we can decrease the false positive rate without significant loss of detection rate. Through extensive experiments conducted with real dataset, our proposed method is proved to be applicable, which obtains the accuracy up to 90% with the false positive rate up to only nearly 5%.

The structure of this paper is organized as follows. Section 2 presents related works, which include victim side DoS detection and source side DoS detection. Section 3 depicts the basic environments of the proposed source side DoS detection by using adaptive threshold method, and describe the main idea of traffic seasonality aware traffic adjustment. In Section 4, the results of extensive evaluation present the viability of the proposed method. Finally, we conclude our paper in Section 5.

## 2. Related Works

Besides many researches which are related to traffic engineering in computer systems such as smart controlling systems, DoS attack detection has also been attracted for a long time. There were various kinds of detection methods such as threshold based method, packet confirmation method, and machine learning based method, and most of them focused on victim-side detection methods. Tsunoda et al. [2] have proposed a DoS attack detection method based on the response packet confirmation mechanism. The basic idea of the confirmation mechanism is that the types of response packets received by a victim could be predicted based on the corresponding types of request packets. If unpredicted packets are observed, an attack is detected. Adversely, there is a need for the router to record the source and destination addresses, protocol fields, etc. for every request it forwards, which requires much memory to execute [26]. Zhang et al. [3] proposed an adaptive sampling technique which is based on the group similarity. This reflects that similar abnormal network traffic could be entities of a group which shares similar characteristics. However, in the

sophisticating attacks, the attackers could mimic the characteristics of legitimate flows, which makes the attack ones be difficult to be detected. In the respect of machine learning based techniques, Gavrilis and Dimitris et al. [13, 14], have proposed some solutions to classify the traffic into attack and normal ones based on different network features. Xiao et al. [21] suggested a usage of correlation analysis between flows in data center to detect Distributed Denial of Service (DDoS) attack. Niyaz et al. [22] have proposed using deep learning with a large set of features derived from network traffic headers to do the classifying work. Being similar to [3], the main drawback of these methods is the attackers could make the attacking network flows or packets be similar to the legitimate ones, which could possibly reduce the detection accuracy. Moreover, if the protected network is large with many servers and network devices, the performance of the network could be limited due to the computation complexity used by the machine learning techniques, especially in [22]. Regarding threshold based DoS attack detection which requires the traffic volume statistics [20], Kawahara et al. [4] have proposed a method for detecting network anomalies by using flow statistics. In which, if the number of sampled flows is larger than a predefined threshold, there would be abnormal events detected. Similarly, YuHunag et al [25] proposed a method which analyzes the frequency of traffic based on OpenFlow [24]. Then if a predefined threshold is exceeded, there will be a notification raised. These methods could be considered to be good with stable traffic condition. However, in the reality, the network traffic could fluctuate significantly. Also, it changes by the time of the day. For example, network traffic could increase when many users becomes more active, while decrease when users go back to sleep.

Meanwhile, there are some efforts which try to situate the attack detection nearer to attack sources. Levy et al. [6] proposed a collaborative method between Internet service providers (ISP). In this method, network traffic will be filtered by one ISP for another ISP, so attack traffic will be removed as soon as it arrives at the first reached ISP. Morrow et al. [12] have proposed a standardization in order to detect DDoS attack, which presents common channels for different domains or groups to exchange information about them. However, these methods could be largely ignored due to the lack of trust [10]. Argyraki et al. [7] proposed a mechanism called *Active Internet Traffic Filtering* (AITF) protocol, which allows AITF-enabled receiver to use the routes recorded on incoming packets. This work is used to find the last point of trust on each attack path, which attack could be detected and blocked there. Undoubtedly, although this method is proved to be effective by the evaluations, deploying it popularly could be a very difficult for the hardware producer due to the performance problem.

Being one of the first research to propose source-side DoS detection, Mirkovic et al. [8, 11] introduced a DoS attack detection mechanism called D-WARD, which is deployed at the exit router of a network. It collects the number of connection and number of destination. Afterward, based on these statistics, the algorithm would find the anomalies in network traffic. Yet, this method requires the support from both victim side and source side, and it could not guarantee that it detects DoS attack with small volume of UDP packets. He et al. [5] proposed a machine learning based method to detect DDoS attack from the source side in a cloud environment. In which, both supervised and unsupervised learning methods are employed to classify network traffic into 2 groups: attack and non-attack. However, in complicating cases, the volume of attack traffic can be very small relatively to legitimate traffic, and the attack traffic can be mixed up to the legitimate traffic, which has been pointed by Deka et al. [21]. Consequently, this subtle attack traffic could cause some problems for machine learning based methods [5]. Up to date, our former work [16] has proposed establishing an adaptive threshold for detecting the subtle DoS attack based on the observed traffic over continuous periods. However, the drawback of this method is high false positive. The main reason is that the process of updating

threshold value is affected by the unknown portion of attack traffic. That is, when attack is detected, it is hard to separate the correct volume of legitimate traffic from the attack traffic, and it is difficult to adjust threshold properly. This raises a need for a solution to adjust the threshold more carefully when attack is detected.

## 3. Traffic Seasonality aware Adaptive Threshold for Source-side DoS Attack Detection

In order to achieve effective source-side DoS attack detection, we propose the traffic seasonality aware adaptive threshold method. The proposed method adjusts the threshold more properly, even though there is attack, by using not only the traffic volume but also traffic seasonality. Especially, the traffic seasonality is obtained by monitoring a source-side network and it is used for separating the legitimate traffic from the attack traffic and updating the threshold for detecting attacks.

First of all, we describe the environment for monitoring and gathering the network traffic of a source-side network where SDN technology is deployed. Subsequently, we describe the basic concept of adaptive threshold approach with fluctuated traffic. Then, we describe how to obtain the seasonality of traffic and how to adjust the threshold with the seasonality of traffic.

### 3.1 Environment for Source-Side DoS Attack Detection

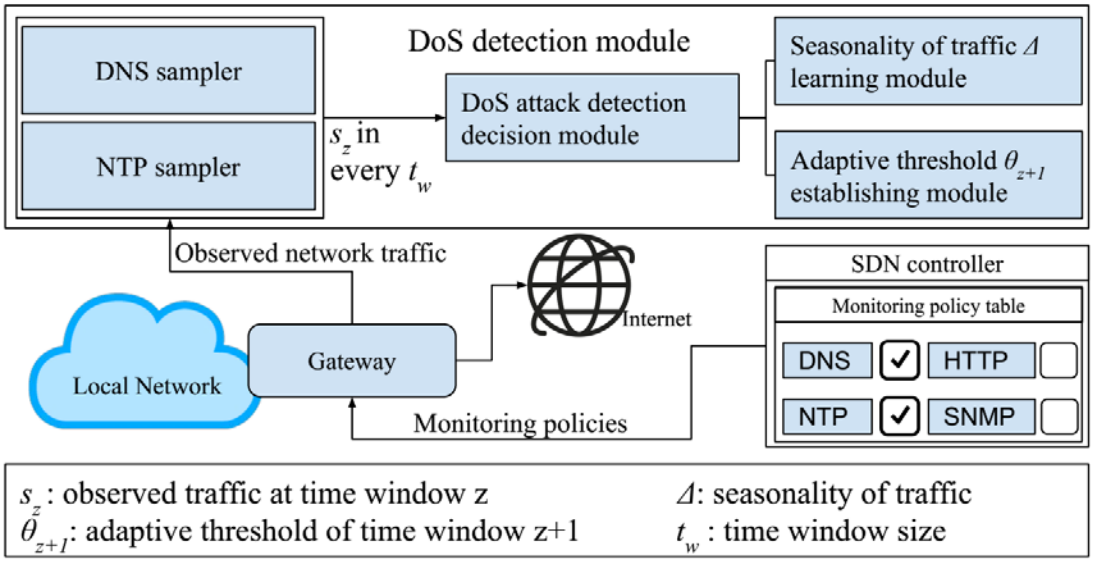



**Fig. 1.** Overall environment of the proposed source-side DoS attack detection

In order to detect DoS attack near sources, the gateway of a local network can be a good location for deploying the functionality of DoS attack detection, especially with IoT devices [15]. **Fig. 1** describes the overall environment of the proposed source-side DoS attack detection. The gateway has functionalities of SDN and the external SDN controller can set the monitoring policy of the gateway properly, by updating the Monitoring policy table. Out of all the network traffic which head from the local network to the Internet, the network traffic related to the specific protocols chosen by the monitoring policy, such as DNS and NTP, is monitored and forwarded to the DoS detection module. That is, in the proposed environment, we can selectively monitor suspicious types of protocols by setting the monitoring policies.

In the Dos detection module, there are traffic samplers corresponding to the chosen protocols. A sampler receives the corresponding forwarded traffic and generates the observed traffic block after each equal duration. We define this equal duration as *time window* and its size is regarded as $t_w$. The observed traffic block during the $z^{th}$ time window is defined as $s_z$. The attack detection decision module detects DoS attack by checking whether the volume of the observed traffic exceeds the threshold or not. If it exceeds the threshold, DoS attack is detected during that time window. The attack detection decision module is supported by two other modules, adaptive threshold establishing module and seasonality of traffic learning module. The adaptive threshold establishing module adjusts the threshold of the next time window, $\theta_{z+1}$, based on the observed traffic of the current time window, $s_z$ (Section 3.2). The seasonality of traffic learning module estimates the volume of normal traffic out of the observed traffic during an attack by using the seasonality of traffic of a given time period, $\Delta$ (Section 3.3).

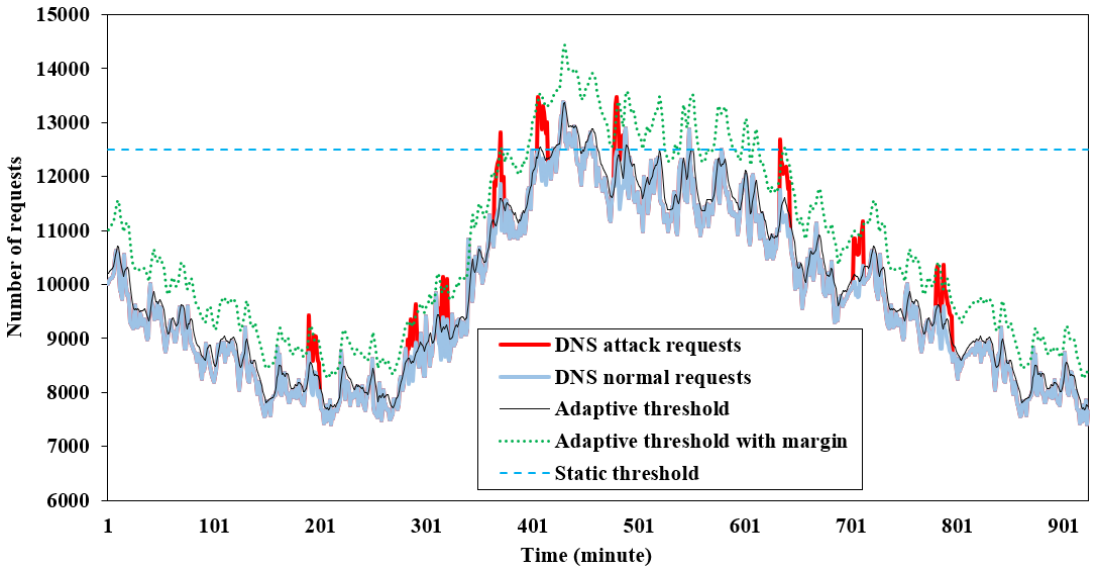



**Fig. 2.** Example of subtle attack traffic near to attack sources and the concept of adaptive threshold

The main objective of the DoS detection module is detecting subtle attack traffic on a heavily fluctuated legitimate traffic. **Fig. 2** depicts an example of subtle attack traffic on legitimate DNS request traffic. The volume of DNS request traffic is significantly changed along to the time and it also has many small legitimate peaks. Up on this heavily fluctuated traffic, some small peaks of attack traffic are mixed up. Because the attack happens near to attack sources rather than a victim and attack sources may be mostly distributed across multiple sub networks, the volume of attack traffic can be very small. In this case, the static threshold method is hard to be applicable, because it can not detect many small attack peaks on the deep valley of the legitimate traffic. Accordingly, we need to adjust the threshold properly to follow the legitimate traffic in order to detect subtle attack peaks. Also, while adjusting threshold, a meaningful margin can be applied in order to prevent false detection caused by fluctuation of legitimate traffic. This threshold adjustment is performed on the adaptive threshold establishing module on every time window, by using exponential smoothing function with the observed traffic in every time window [17].

By adjusting the threshold with smoothing function, we can make the threshold follow the legitimate traffic well, unless spontaneous attack peaks happen. When attack happens, the

observed traffic in a time window possesses both legitimate and attack traffic. In this case, we need to separate the legitimate traffic from the attack traffic in order to adjust the threshold correctly. However, it is very difficult to separate these traffic correctly in realtime, and we try to estimate the portion of legitimate traffic by considering the seasonality of traffic. That is, by observing the trend of changes of traffic in a relatively long time, we learn the likelihood of traffic changes in a given time window and predict the volume of legitimate traffic. This process is performed on the seasonality of traffic learning module. With the estimated volume of legitimate traffic, we adjust the threshold more conservatively.

### 3.2 Traffic Volume aware Adaptive Threshold Establishment

In the adaptive threshold establishing module, the threshold for the next time window, $\theta_{z+1}$, will be calculated by using exponential smoothing function with the observed traffic in the current time window, $s_z$. Also, the threshold is leveraged by using the margin $\delta$.

Basically, the threshold value is calculated by the forecasted traffic of the next time window. Consider at the $z^{th}$ time window, we have the forecasted traffic $\bar{s}_z$ which is provided at the $(z-1)^{th}$ time window, and obtain the observed traffic, $s_z$. Regarding the exponential smoothing function, the forecasted traffic in the $(z+1)^{th}$ time window, $\bar{s}_{z+1}$, is calculated as :

$$\bar{s}_{z+1} = \mu \ s_z + (1 - \mu) \ \bar{s}_z \qquad (1)$$

In which, the coefficient $\mu$ ($0 \leq \mu \leq 1$) is a smoothing parameter. The smaller $\mu$ is, the more weight the forecasted traffic of the $z^{th}$ time window would contribute to the next one. On the other hand, the higher $\mu$ is, the more weight the current observation would contribute to the next one. In this paper, we keep the value of this coefficient as 0.5.

Because this forecasting takes the prediction from the previous time window as input, there is a need for assigning the predicted value of the first time window, $\bar{s}_1$ because there is no previous value before it. We assign this initial value as the number of request in the first time window as :

$$\bar{s}_1 = s_1 \qquad (2)$$

Then we would have following series of equations for forecasted traffic.

$$\bar{s}_2 = \mu \ s_1 + (1-\mu) \ \bar{s}_1$$
$$\dots$$
$$\bar{s}_{z+1} = \mu \ s_z + (1-\mu) \ \bar{s}_z$$

Consequently, $\bar{s}_{z+1}$ can be calculated under the regression form as follow.

$$\bar{s}_{z+1} = \sum_{i=0}^{z-1} \mu \ (1-\mu)^i \ s_{z-i} + (1-\mu)^z s_1 \qquad (3)$$

With this forecasted traffic, we may set the threshold and the threshold can follow the change of traffic in the aspect of long term. Yet, there can be local fluctuation of traffic and these fluctuated traffic may exceed the threshold accidentally, and it causes misdetection and false positives. To prevent this unwilling false positives, we can provide a meaningful gap between the forecasted traffic and the threshold by using a margin value $\delta$ ($\delta \geq 0$). This margin value is a knob parameter of this detection method. With higher margin value, the detection

method is more tolerable to local fluctuations, but it may miss some attack peaks. Detail impacts of the margin value is presented on Section 4.

Accordingly, the threshold of $(z+1)^{th}$ time window is calculated as follow:

$$\theta_{z+1} = \bar{s}_{z+1} * (1+\delta) \quad (4)$$

Subsequently, this adjusted threshold is provided to attack detection decision module for the $(z+1)^{th}$ time window, and if the observed traffic $s_{z+1}$ exceeds the threshold, the attack is detected.

This threshold adjustment based on the observed traffic could not work well if there is attack traffic in the observed traffic. When attack is detected at the $z^{th}$ time window, the observed traffic $s_z$ should not be used for establishing the threshold of the $(z+1)^{th}$ time window because it contains both the attack and legitimate traffic. Specifically, if the observed traffic contains attack traffic and it is used for calculating the forecasted traffic with equation (1), the forecasted traffic and the threshold of the next time window would be overestimated. As a result, some subtle attacks can not be detected with this overestimated threshold. Accordingly, to update the threshold properly under attack, we use the seasonality of traffic for estimating the portion of legitimate traffic out of the observed traffic which includes attack traffic, which is described more specifically in the following subsection.

### 3.3 Traffic Seasonality aware Threshold adjustment

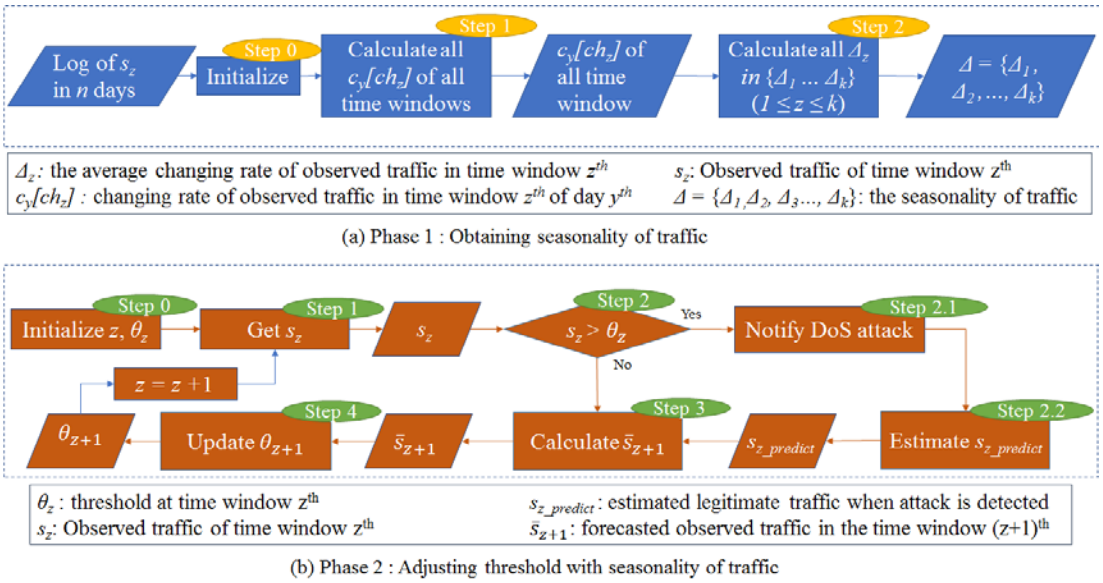



**Fig. 3.** The overall procedure of DoS detection with traffic seasonality aware threshold adjustment

The basic idea of traffic seasonality aware threshold adjustment is updating threshold by using the estimated legitimate traffic which is calculated with the seasonality of traffic even though there is attack. In the proposed method, the seasonality of traffic is considered as the changing rate of the traffic of a corresponding time window of a day.

In a network environment, we may observe the natural seasonality of traffic. For example, the observed traffic of DNS protocol in a company could increase much in the morning, and it decreases gradually from the afternoon until the evening. Accordingly, if we could capture the seasonality, i.e. changing trend, of traffic on a specific time, we may estimate the volume of

traffic on the corresponding time with imperfect traffic information such as the observed traffic mixed with both legitimate and attack traffic. Also, this seasonality is different to every distinct network, because the behavior of user or system of the network is different to each other. It is the main reason that we need to provide a valid method for obtaining the seasonality of traffic in a network.

**Fig. 3** illustrates the overall procedure of DoS detection with traffic seasonality aware threshold adjustment. There are two phases : 1) obtaining seasonality of traffic and 2) adjusting threshold with seasonality of traffic. In the first phase, the seasonality of traffic, i.e. the changing rate between two continuous time windows, is calculated by using the traffic logs. The second phase includes how to calculate the estimated legitimate traffic by using the seasonality of traffic even under attack situation and how to update threshold with the forecasted traffic which is calculated with the estimated legitimated traffic. Details of each phase are provided subsequently in the following subsections.

### 3.3.1 Obtaining the seasonality of traffic

Because the threshold is updated on every time window, we model the seasonality of traffic in the aspect of time windows. That is, we learn the trend of traffic change on every time window as the seasonality of traffic.

Let us assume that there are $n$ days legitimate traffic logs, $D = \{d_1, d_2,...,d_n\}$ and there are $k$ time windows per a day. The actual value of $k$ is related to the length of time window. If we set the minimum value of the length of time window is one minute, $k$ can be calculated as $1440/t_w$, where $t_w$ is the length of time window. Afterward, we can define a set of observed traffic $s_z$ ($1 \leq z \leq k$) for the $y^{th}$ day as follows :

$$d_y = \{s_1, s_2 \ldots s_k\} \tag{5}$$

In here, $d_y[s_z]$ is used to present $s_z$ in $d_y$, that is, the observed traffic on the $z^{th}$ time window of the $y^{th}$ day. With this data, we obtain the changing rate of observed traffic between 2 continuous time windows for each day and define it as follow :

$$c_y = \{ch_1, ch_2 \ldots, ch_k\} \tag{6}$$

In here, $ch_z$ ($1 \leq z \leq k$) stands for the changing rate corresponding to the $z^{th}$ time window of a day, and $c_y$ is the set of changing rate for the $y^{th}$ day. $c_y[ch_z]$ stands for the changing rate corresponding to the $z^{th}$ time window on day $y^{th}$, and it is calculated as follow :

$$c_y[ch_z] = \begin{cases} \dfrac{d_y[s_1]}{d_{y-1}[s_k]}, z = 1 \\ \dfrac{d_y[s_z]}{d_y[s_{z-1}]}, z > 2 \end{cases} \tag{7}$$

As we assumed that there are $n$ days logs, we can get a set of changing rate $C = \{c_1, c_2, \ldots, c_n\}$. In here, we calculate the average value of all changing rate corresponding to a same time windows for every day in order to obtain the trend of traffic change on a time window. We define the average changing rate at the $z^{th}$ time window as $\Delta_z$, and calculate it as follow :

$$\Delta_z = \frac{\sum_{y=1}^{n} c_y[ch_z]}{n}, 1 \leq y \leq n, 1 \leq z \leq k \tag{8}$$

**Algorithm 1**: Obtaining Seasonality of Traffic, Δ
**Input**: $D = \{d_1, d_2 \ldots, d_n\}$, $d_y = \{s_1, s_2 \ldots s_k\}$, $t_w$
**Output**: $\Delta = \{\Delta_1, \Delta_2, \Delta_3 \ldots, \Delta_k\}$
// Step 0: Initialization
1. **Initialize**
2. **For** $y = 1$ **to** $n$ **do**
3. $c_y = \{ch_1 = 0, ch_2 = 0 \ldots, ch_k = 0\}$ // assign all change as 0 initially
4. $\Delta = \{\Delta_1 = 0, \Delta_2 = 0 \ldots, \Delta_k = 0\}$ //initialize Δ as 0

// Step 1: Calculating the changing rate of observed traffic in all time windows
5. **For** $y = 1$ **to** $n$ **do** //loop for all n days
6. **For** $z = 1$ **to** $k$ **do** // loop for all k time windows
7. **If** ($z = 1$) **then** // If the index is the first time window
8. **If** ($y$ != 1) **then** // if the considered day is not the first day, calculate the delta
9. $c_y[ch_z] = \frac{d_y[s_z]}{d_{y-1}[s_k]}$
10. **Else** $c_y[ch_z] = 0$
11. **Else** $c_y[ch_z] = \frac{d_y[s_z]}{d_y[s_{z-1}]}$
12. $z = z + 1$
13. $y = y + 1$

// Step 2: Averaging the changing rate of observed traffic in all time windows
14. **For** $z = 1$ **to** $k$ **do** //loop for all k time windows
15. **If** ($z$ != 1) **then** $\Delta_z = \frac{\sum_{y=1}^{n} c_y[ch_z]}{n}$ //sum all $c_y[ch_z]$ from n days then take average
16. **Else** $\Delta_z = \frac{\sum_{y=2}^{n} c_y[ch_z]}{n-1}$ //the first day's first time window does not have previous one.

According to this calculation, we can get a set of average changing rate of a day, $\Delta = \{\Delta_1, \Delta_2, \Delta_3 \ldots, \Delta_k\}$. This calculation is performed on the seasonality of traffic learning module and on every time window $\Delta$ is updated. The entire algorithm for obtaining the seasonality of traffic is presented in **Algorithm 1**. The input of the algorithm is a traffic log of $n$ days, which contains the observed traffic of all time windows with a given corresponding size, and the output is $\Delta = \{\Delta_1, \Delta_2, \Delta_3 \ldots, \Delta_k\}$. The algorithm is composed of three steps : 1) Initialization (Step 0), Calculating the changing rate of observed traffic in all time windows (Step 1), and Averaging the changing rate of observed traffic in all time windows.

- Step 0 : All the average changing rate of a day $\Delta_z$ ($1 \leq z \leq k$) and the changing rate of any time window on each day $c_y[ch_z]$ ($1 \leq y \leq n$ and $1 \leq z \leq k$) are initialized as 0.
- Step 1 : All the changing rate of the $z^{th}$ time window on the $y^{th}$ day ($1 \leq y \leq n$ and $1 \leq z \leq k$) are calculated by using equation (7). The value of $c_1[ch_1]$ is assigned with 0 because there is no previous time window.
- Step 2 : The average of changing rate $\Delta_z$ ($1 \leq z \leq k$) of a time window in a day is calculated by equation (8). For the first time window ($z = 1$), the average value is not divided by $n$, but by ($n$-1) because the value of $c_1[ch_1]$ is assigned with 0 as mentioned in Step 1.

The complexity of the algorithm 1 is mainly related to the number of days, *n*, and the total number of time window of a day, *k*. In the algorithm 1, each step has two nested loops which takes *k* and *n* entities for each. Therefore, the time complexity for obtaining the seasonality of traffic is O(3**n***k*) and it can be simplified to O(*n***k*). In the aspect of the space complexity of the algorithm 1, (*n***k*) values of $d_y[s_z]$, (*n***k*) values of $c_y[ch_z]$ and *k* values of $\Delta_z$ ($1 \leq y \leq n$ and $1 \leq z \leq k$) are stored for calculation. Consequently, the space complexity for obtaining the seasonality of traffic is O(2**n***k*+*k*) and it can be simplified to O(*n***k*).

### 3.3.2 Threshold adjustment with seasonality of traffic

As we mentioned earlier, the seasonality of traffic is used for estimating the volume of legitimate traffic out of the observed traffic under attack. If there is no attack at the $z^{th}$ time window, the forecasted traffic of the $(z+1)^{th}$ time window, $\bar{s}_{z+1}$, is calculated with the current observed traffic, $s_z$, by using equation (1) and the corresponding threshold is updated by using equation (4). However, if an attack is detected at the $z^{th}$ time window, we calculate the forecasted traffic, $\bar{s}_{z+1}$, with the estimated legitimate traffic, $s_{z\_predict}$ instead of the current observed traffic, $s_z$, as follow :

$$\bar{s}_{z+1} = \mu \ s_{z\_predict} + (1 - \mu) \ \bar{s}_z \tag{9}$$

In here, $s_{z\_predict}$ is calculated by multiplying the changing rate $\Delta_z$ of the corresponding $z^{th}$ time window to the observed traffic of the previous time window, $s_{z-1}$, which is considered to contain only legitimate traffic, as follow:

$$s_{z\_predict} = s_{z-1} \ * \Delta_z \tag{10}$$

$\Delta_z$ means average traffic changing rate from the $(z-1)^{th}$ time window to the $z^{th}$ time window. And we need to make sure that there is no attack on the $(z-1)^{th}$ time window. After calculating the forecasted traffic with the estimated legitimated traffic by using the seasonality of traffic, the corresponding threshold is updated by using equation (4).

If the duration of an attack is longer than the length of a time window, and the attack spans over multiple time windows. In this case, we can not use the observed traffic of the previous time window (i.e. $s_{z-1}$) for calculating the estimated legitimate traffic (i.e. $s_{z\_predict}$), because the observed traffic contains unknown amount of attack traffic. Accordingly, we use the estimated legitimate traffic sequently for calculating the estimated legitimate traffic of the next time window as equation (11).

$$s_{z\_predict} = s_{(z-1)\_predict} \ * \Delta_z \tag{11}$$

Specifically, if there is no attack on the $(z-2)^{th}$ time window and there are attacks on the $(z-1)^{th}$ and the $z^{th}$ time windows continuously, $s_{(z-1)\_predict}$ is calculated by using equation (10) with the values of $s_{z-2}$ and $\Delta_{z-1}$ and $s_{z\_predict}$ is calculated by using equation (11). This successive calculation of the estimated legitimate traffic with equation (11) will stop only if there is no more attack traffic is detected.

**Algorithm 2 – STBAT algorithm**: DoS detection with seasonality aware threshold adjustment

**Input**: $\Delta = \{\Delta_1, \Delta_2 \ldots, \Delta_k\}$, Margin $\delta$, Smoothing parameter $\mu$, Time window length $t_w$

**Output**: Attack notification

1. **Initialize**: //Step 0 : Initialzation
2. $z = [\frac{curren\ time\ in\ minutes}{t_w}] + 1$
3. $\theta_z = \infty$
4. $cAttack = 0$
5. **While**(true)
6. $s_z = getObservedTraffic(z)$ //Step 1 : Traffic Observation
7. **If** (*z is the first time window when the algorithm starts*) **then**
8. $\bar{s}_z = s_z$
9. **If** ($s_z > \theta_z$) **then** //Step 2 : Attack Detection
10. *NotifyDoSAttack(z)* //Step 2.1 : Attack Notification
11. **If** ($cAttack == 0$) **then** $s_{z\_predict} = s_{z-1} * \Delta_z$ //Step 2.2 : Estimated Legitimate
12. **Else** $s_{z\_predict} = s_{(z-1)\_predict} * \Delta_z$ Traffic Calculation
13. $\bar{s}_{z+1} = \mu * s_{z\_predict} + (1 - \mu) * \bar{s}_z$ //Step 3 : Forecasted Traffic
14. $cAttack$ ++ Calculation under attack
15. **Else**
16. $\bar{s}_{z+1} = \mu * s_z + (1 - \mu) * \bar{s}_z$ //Step 3 : Forecasted Traffic
17. $cAttack = 0$; Calculation under no attack
18. $\theta_{z+1} = \bar{s}_{z+1} * (1 + \delta)$ //Step 4 : Threshold Update
19. $z = z + 1$
20. **If** ($z > \frac{1440}{t_w}$) **then** $z = 1$

Algorithm 2 describes the algorithm of detecting DoS attack with traffic seasonality aware threshold adjustment, which is called STBAT (Seasonality Traffic Behavior Aware Threshold). In this algorithm, we set the minimum time window length as one minute, and z is set initially by using the current time in minutes (line 2). For example, if the algorithm is started at 8:00 am and the time window size is 3 minutes, the considered current time in minutes is 480 and z is initialized as 161 (480/3 + 1). Also, the initial threshold sets as infinity and there is no attack happens on the first time window (line 3). After this initialization process (Step 0), on every time window, traffic observation (Step 1), attack detection (Step 2), attack notification (Step 2.1), estimated legitimate traffic calculation (Step 2.2), forecasted traffic calculation (Step 3), and threshold update (Step 4) are executed as follows :

- Step 1: The observed traffic $s_z$ is obtained by the function *getObservedTraffic(z)* (line 6).
- Step 2: The observed traffic $s_z$ is compared with the thereshold $\theta_z$. If it exceeds the threshold, an attack is detected on the corresponding time window (line 9).
    - Step 2.1: If an attack is detected, the attack is notified by the function *NotifyDoSAttack(z)* (line 10).
    - Step 2.2: If an attack is detected, the estimated legitimate traffic $s_{z\_predict}$ is calculated with the seasonality. If the attack spans over multiple time windows ($cAttack \neq 0$), the estimated legitimate traffic is calculated by using equation (11). Otherwise, it is calculated by using equation (10) (line 11 and line 12).

- Step 3: The forecasted traffic $\bar{s}_{z+1}$ of the next time window $(z+1)^{th}$ is calculated. If no attack is detected, the threshold is updated by using the equation (1) (line 16). Otherwise, if any attack is detected, the threshold is updated by using the equation (9) (line 13).
- Step 4: The threshold $\theta_{z+1}$ of the next time window $(z+1)^{th}$ is calculated using equation (4) with the margin value $\delta$ (line 18).
- Subsequently, the algorithm will continue for the next time window. If the z value exceed the limit of a day, z reset to 1 for the next day (line 19-20).

The algorithm 2 executes the main loop (from line 6 to line 20) continuously on each time window. Because there is no loop inside this main loop, the time complexity of algorithm 2 is simply O(1). In the aspect of the space complexity, the seasonality representatives, that is, $\Delta = \{\Delta_1, \Delta_2 \ldots, \Delta_k\}$, and some other variables inside the main loop are stored during the algorithm is executed. Consequently, the space complexity of the algorithm 2 is simply O(k).

## 4. Evaluation

In order to evaluate the viability of the proposed traffic seasonality aware adaptive threshold algorithm, called STBAT, for source-side DoS detection, we evaluated STBAT in the aspects of detection rate and false positive rate, primarily in comparison to different adaptive threshold based DoS detection algorithms. The detection rate is the likelihood that a detection method detects attacks correctly in a time window. The false positive rate is the likelihood that a detection method considers a legitimate traffic as an attack in a time window. As the definition, high detection rate and low false positive rate is preferred to be a better detection method.

### 4.1 Evaluation settings

***Data sets.*** For evaluating the proposed algorithm, we choose DNS query traffic as input traffic of the proposed system. DNS server is mostly an easy target of DoS attack, and DNS protocol itself can be used as DDoS attack as well as DRDoS (Distributed Reflection DoS) attack. In order to evaluate the performance of the proposed algorithm in a real situation, we collect DNS query traffic from DNS-STAT:Hedgehog [18,19] which gathers the real time statistics for DNS traffic in the world. DNS-STAT:Hedgehog provides the average number of queries per second in a single time window such as one minute and five minutes, and we can gather a traffic log which presents the number of queries per time window. And this traffic information is provided by different countries and different query types. In this evaluation, we collect DNS query traffic of South Korea region for 2018 January, Japan region for 2018 May, USA region for 2018 May with one-minute time window. During collecting DNS query traffic, we exclude the major outliers which might be related to previous DoS attack events, then we consider the collected DNS query traffic as legitimate traffic.

***Attack emulation.*** In order to understand how the proposed algorithm reacts to various attack situations, we generate subtle attack peaks on the last day of the collected DNS query traffic on every evaluation. To generate a subtle attack peak, we randomly pick a time window of the traffic and increase the number of queries corresponding to the selected time window. In here, we can set the attack rate and the attack duration. The attack rate is the portion of number of attack queries over the legitimate number of queries at the corresponding time window. The attack duration is the length of continuous time windows where attack happens. In this

evaluation, we vary attack rate from 8% to 12% to generate subtle attack peaks, and we vary attack duration from 1 minute to 15 minutes. In this evaluation, we keep the total length of attacks to 90 minutes. That is, if we use 1 minute attack duration, 90 distinct attack events are generated. On the other hand, if we use 5 minutes attack duration, 15 distinct attack events are generated.

*Compared Algorithms.* Because the main objective of this evaluation is understanding the impact of the traffic seasonality aware threshold adjustment, we mainly compare the proposed algorithm, STBAT, to the different adaptive threshold based DoS detection algorithms including OTAT and TB.

OTAT algorithm means using the traffic volume aware adaptive threshold establishment without help of seasonality aware threshold adjustment, and OTAT is analogous to the previous approach in [16]. In OTAT, when attack is detected at a time window, it keeps the threshold value because it cannot guarantee how much portion of observed traffic is legitimate traffic. OTAT update the threshold only if there is no attack is detected on a time window. Specifically, when an attack is detected at the $z^{th}$ time window, OTAT algorithm does not use the observed traffic $s_z$ for establishing the threshold of the $(z+1)^{th}$ time window, $\theta_{z+1}$, and it handles this case by keeping the threshold until no attack is detected. After this attack detection, continuous attacks are detected up to the $(q-1)^{th}$ time window ($z < q$) and no attack is detected at the $q^{th}$ time window, the threshold at the $(q+1)^{th}$ time window is updated by using the traffic estimation as follows $\bar{s}_{q+1} = \mu * s_q + (1 - \mu) * \bar{s}_z$.

TB algorithm means using the traffic volume aware adaptive threshold establishment with time-based threshold updating method. TB algorithm assumes that an attack lasts on a specific duration with length $t_a$. And if an attack is detected at a time window, the threshold is kept for the next continuous time windows until no more attack is detected on a time window or the duration of detecting the attack exceeds $t_a$. Specifically, when an attack is detected at the $z^{th}$ time window, TB algorithm does not update the threshold of the $(z+1)^{th}$ time window, $\theta_{z+1}$. After this attack detection, if continuous attacks are detected until the $p^{th}$ time window ($z < p < z + t_a$), the threshold is kept up to the $p^{th}$ time window. Then, at the $(p+1)^{th}$ time window, the threshold is updated by using the traffic volume aware adaptive threshold establishment. Otherwise, if the duration of the attack is longer than $t_a$, the threshold is updated at the $(z+t_a+1)^{th}$ time window with the observed traffic of this time window, $s_{z+t_a+1}$. Afterwards, the threshold is updated by the traffic volume aware adaptive threshold establishment.

In these adaptive threshold-based algorithms, the margin, $\delta$, is used in the same way. That is, margin means the leveraging portion to the estimated traffic in order to mitigate the impact of arbitrarily fluctuated legitimate traffic. In this evaluation, we use margin value from 2 to 10.

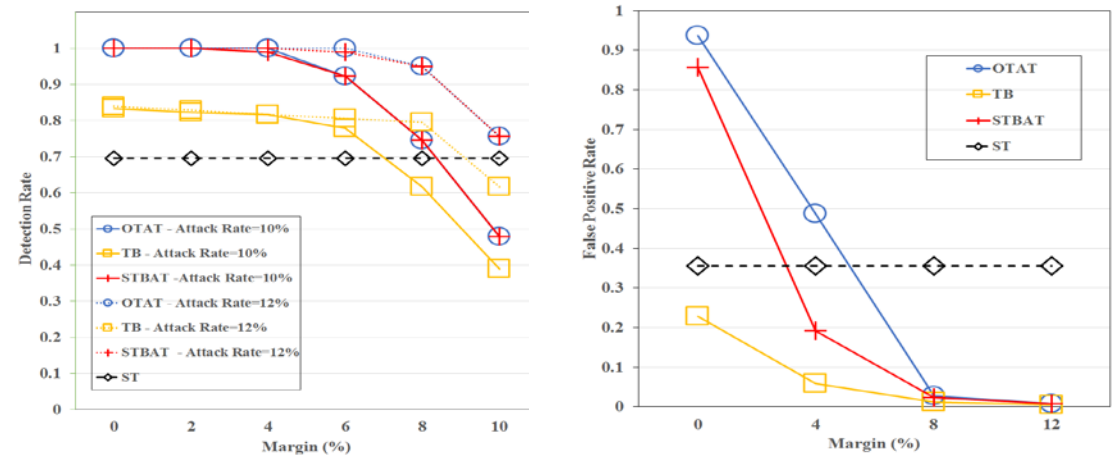
Also, we compare STBAT to the static threshold approach, ST, which is mostly used in victim-side DoS detection methods. In this evaluation, the static threshold is set with the average volume of entire traffic with 10% margin value.

## 4.2 Evaluation results

### 4.2.1 Impact of traffic seasonality aware threshold adjustment with different margin

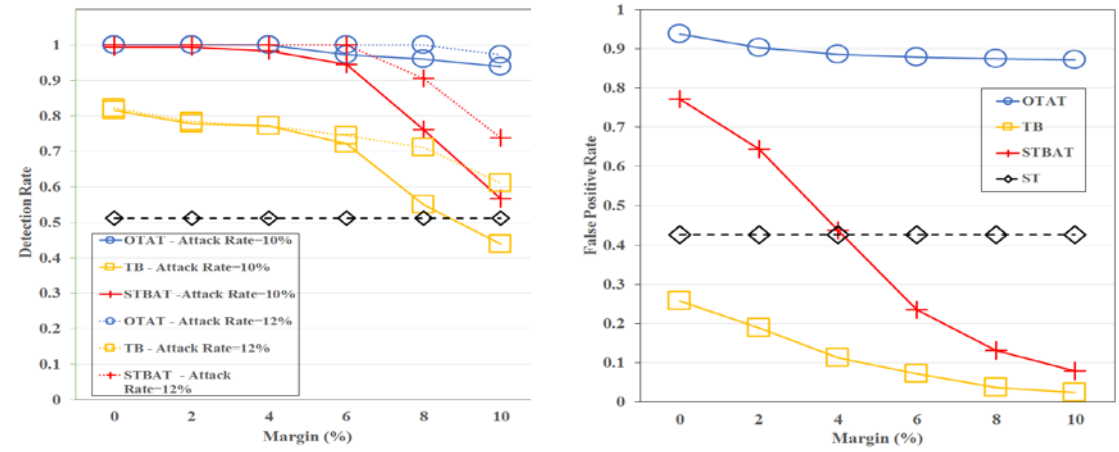
In the first evaluation, which the results are shown in **Fig. 4**, **Fig. 5** and **Fig. 6**, we mainly evaluate the impact of traffic seasonality on detection rate and false positive rate with different margin. We employ OTAT, STBAT and TB algorithms with different margin values to compare the performance. For ST algorithm, we keep 10% margin value. We keep the time

window size as 1 minute and keep the attack duration as 1 minute. For TB algorithm we keep $t_a$ as 1 minute as well in order to do fair comparison. And we use two different attack rate 10% and 12%. In here, we observed that the attack rate significantly affects to the detection rate but it does not affect the false positive rate. That is, we employ a single line for all attack rate with the corresponding algorithm in **Fig. 4 (b)**, **Fig. 5 (b)** and **Fig. 6 (b)**.



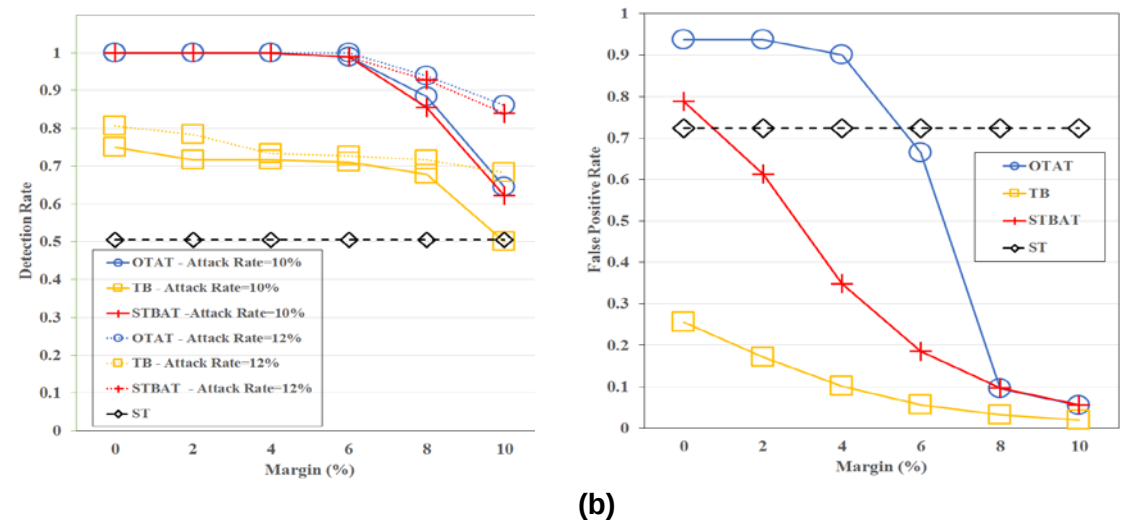

(a) (b)

**Fig. 4.** Detection rate and false positive rate with South Korea dataset




(a) (b)

**Fig. 5.** Detection rate and false positive rate with Japan dataset




(a) (b)

**Fig. 6.** Detection rate and false positive rate with USA dataset

First of all, we can easily observed that ST, static threshold approach is not applicable to source-side DoS detection where subtle attack peaks happen frequently. ST does not consider the fluctuation of legitimate traffic carefully, and it misses many attack peaks located in the valley of the fluctuated legitimate traffic. Also, ST considers many local peaks of legitimate traffic as attack and its false positive rate becomes high.

Among the adaptive threshold based algorithms, STBAT achieves high detection rate and reasonable false positive rate on every dataset. We observed that STBAT and OTAT achieve similar high detection rate on South Korea and USA dataset. In Japan dataset, STBAT and OTAT have similar detection rate when the margin value is less than 8%. However, STBAT has much lower false positive rate than OTAT in every dataset. For TB, it achieves very low false positive rate, but the detection rate also becomes low.

In TB, the threshold is updated in more optimistic manner than STBAT and OTAT. That is, regardless of the traffic observation and estimation after an attack, it updates the threshold optimistically. Then, the adaptive range of the threshold of TB is little higher than STBAT and OTAT. Through analyzing the input dataset, we found that the higher volume traffic after detecting an attack makes TB keeps higher range of the threshold. The followings are some specific cases for observing higher volume traffic after $t_a$:

- Long-lasting attack : An attacker may launch an attack longer than $t_a$. In this case, the continuous attack just after $t_a$ which contains high volume of traffic is used for establishing the threshold and the threshold becomes higher than before. Consequently, the continuous attacks may not detected as an attack anymore.
- High volume traffic after false positive: Though a false positive case happens, TB updates the threshold anyway. Usually, a false positive case occurs when the legitimate traffic increases. That is, the followed traffic after a false positive case may be high volume traffic. In this case, the updated threshold by TB becomes higher. Consequently, the possibility of detecting attacks may decreases.

Accordingly, it decreases the chance of mistaking an legitimate traffic peak as an attack peak. However, this optimistic approach makes TB miss some attack peaks which are more subtle and the detection rate becomes low.

On the other hand, OTAT updates the threshold in more strict manner and it suppresses the threshold in order to detect more subtle attack peaks. OTAT only updates the threshold properly when there is no attack and whenever an attack is detected, even though a false positive case occurs, OTAT the threshold is kept. That is, the adaptive range of the threshold of OTAT becomes little lower than STBAT and TB. Accordingly, OTAT achieves very high detection rate with the suppressed threshold. Especially, in the Japan dataset, it achieves the highest detection rate among the compared algorithms which is more than 90% regardless of margin value. However, the threshold is suppressed too aggressively and it increases the chance of mistaking an legitimate traffic peak as an attack peak. That is, OTAT achieves high false positive rate. Especially, in Japan dataset, OTAT achieves near 90% of false positive rate. Consequently, even though OTAT achieves high detection rate, the system is hampered by high false positive rate. Through the analysis of the input dataset, we noticed that substantial increase of legitimate traffic causes false positive cases in OTAT. That is, if the increment of traffic becomes bigger than the margin, OTAT considers the following traffic as an attack anyway and the keep the threshold as low as possible. Then, the following traffic is considered as attack traffic continuously and false positive cases keep happening.

In STBAT, the threshold is adjusted based on the change of traffic volume as well as the estimated legitimate traffic volume by considering the seasonality of traffic. That is, STBAT updates the threshold more carefully, especially, when an attack event is detected. Because STBAT separates the legitimate traffic from the attack traffic during an attack is detected, even though it is a false positive case, it updates the threshold more precisely than other algorithms. According to this, STBAT can achieve high detection rate which is similar to OTAT, and it achieves lower false positive rate than OTAT with proper margin value.

As we can see in **Fig. 4**, **Fig. 5** and **Fig. 6**, the margin value is the important parameter for the performance of DoS detection in the adaptive threshold algorithms, even though for STBAT. In **Fig. 5**, when the value of margin is 2%, it achieves almost 100% of detection rate but it achieves around 60% of false positive rate. On the other hand, when the value of margin is 10%, the false positive rate of STBAT drops down to around 10%, but the detection rate of STBAT also drops down around 60%. This tradeoff of performance caused by the value of margin can be observed in every dataset. That is, with smaller value of margin, higher detection rate can be achieved but false positive becomes higher. And with the bigger value of margin, the false positive rate keeps lower but the detection rate also becomes lower. According to this tradeoff of performance, it is essential to set proper margin value for adaptive threshold based algorithms. In each dataset, the proper margin value is different to each other. As observed through the figures, the proper margin value is 4%, 6%, and 6% for Korea dataset, Japan dataset, and USA dataset, respectively. More detail evaluation of the impact of margin is described in the subsection 4.2.2.

The attack rate mainly affects to the detection rate of all algorithms. When the attack rate increases from 10% to 12%, we noticed that the detection rate also increases for all algorithms as shown in **Fig. 4 (a)**, **Fig. 5 (a)** and **Fig. 6 (a)**. Especially, when margin value is bigger, the impact of the attack rate becomes bigger. In **Fig. 4(a)**, when the margin is 10% and the attack rate changes from 10% to 12%, the detection rate of STBAT changes from 48% to 76%. On the other hand, we observed that the false positive rate is not affected significantly by the attack rate as shown in **Fig. 4 (b)**, **Fig. 5 (b)** and **Fig. 6 (b)**. Accordingly, we noticed that the attack rate is only applicable to the individual attack detection step and the overall threshold changes may not be affected by the attack rate significantly. More detail evaluation of the impact of attack rate is described in subsection 4.2.3.

### 4.2.2 Relationship between margin and false positive

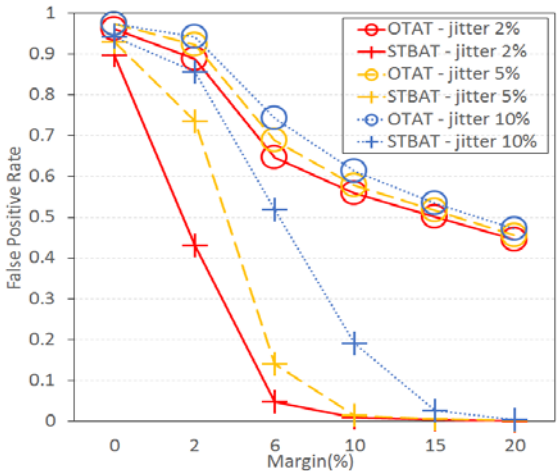



**Fig. 7.** Impact of jitter degree on false positive rate with different margin values

Interestingly, we observe that the false positive rate is not affected by the attack rate, but mainly affected by the margin value. For better understanding the relationship between the margin and the false positive rate, we evaluate the impact of the local fluctuation of legitimate traffic to the false positive rate with different margin values, as shown in **Fig. 7**. We define the portion of amount of local fluctuation to the average of legitimate traffic as jitter degree of the traffic. Specifically, if there is a legitimate traffic with jitter degree $j$ and the average amount of legitimate traffic is $s_z$ at the $z^{th}$ time window, the observed legitimate traffic of the $z^{th}$ time window can be in the range of $[s_z - \frac{s_z * j}{100}, s_z + \frac{s_z * j}{100}]$.

In **Fig. 7**, we observed that bigger jitter degree causes more false positives in both OTAT and STBAT. And we also noticed that the margin value can be set by monitoring the false positive rate of STBAT algorithm. For example, the proper margin value for legitimate traffic with jitter degree of 2%, 5% and 10% can be 6%, 10% and 15%, respectively. Accordingly, we can set the desired level of false positive rate such as 0.1, then we can set the margin value adaptively on every time window.

Unlike STBAT which uses traffic seasonality aware threshold adjustment, OTAT can not reduce the false positive rate less than 0.4 with 20% margin value. This result shows that using seasonality aware threshold adjustment is essential for source-side DoS detection method.

### 4.2.3 Impact of traffic seasonality aware threshold adjustment with different attack types

Attackers may vary attack rate and attack duration in practice, we evaluate the performance of the compared algorithms with different types of attacks. At first, we evaluate the impact of the different attack rate to the detection algorithms as shown in **Fig. 8**. In this evaluation, we keep the attack duration as 1 minute and time window size is kept as 1 minute.

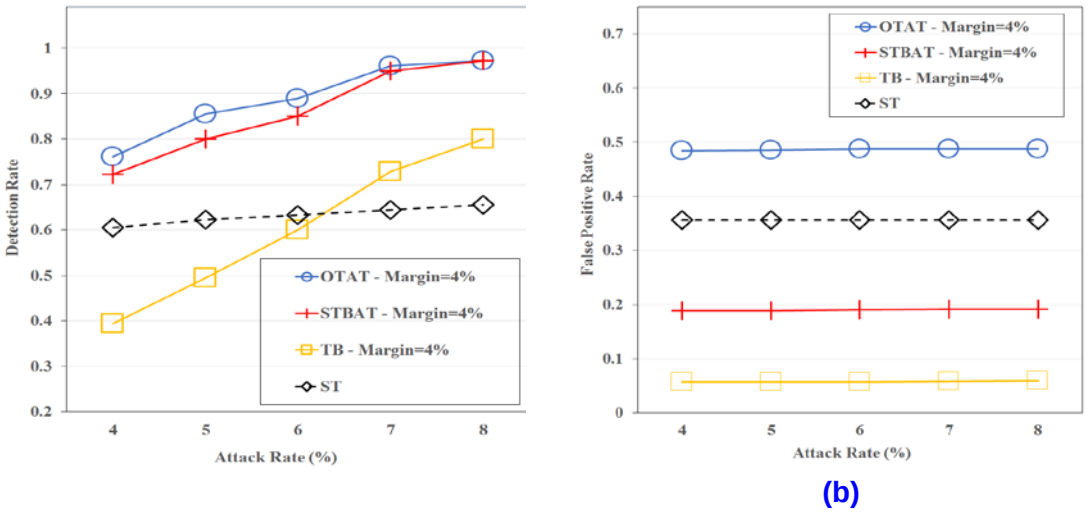



**Fig. 8.** Impact of attack rate on detection rate and false positive rate

At first, we noticed that the detection rate increases along with the attack rates as shown in **Fig. 8(a)**. That is, if attack rate is very small, the detection rate becomes low. Especially, in the case of TB, the detection rate is affected significantly by the attack rate, and with 4% attack rate, detection rate drops around 0.4. OTAT and STBAT are also affected by the small attack rate, and 0.77 and 0.73 detection rates are obtained, respectively. When attack rate becomes smaller, the peaks of attack traffic becomes more similar to the local peaks of legitimate traffic. Then, if we adjust the threshold more optimistically such as TB, the detection rate becomes very low. Otherwise, if we update the threshold more strictly such as OTAT, the decrease of detection rate caused by the small peaks of attack traffic is limited. Also, STBAT, which update the threshold with the consideration of seasonality, achieves similar detection rate to OTAT.

On the other hand, in **Fig. 8(b)**, OTAT has much more false positives than STBAT and TB. That is, even though OTAT achieves the highest detection rate, it is meaningless because of the highest false positive rate. That is, STBAT is the best choice for source-side DoS detection. Also, we could see that false positive rate of detection algorithms are not affected by different attack rate as we described as previous subsection.

In **Fig. 9**, we present the evaluation results of the impact of attack duration to adaptive threshold based DoS detection algorithms. In this evaluation, we set the margin as 4% and attack duration varies from 1 minutes to 15 minutes. As shown in **Fig. 9(a)**, detection rate increases along with the attack duration. That is, more attack traffic is exposed in the legitimate traffic, the volume of attack traffic becomes more significant and any detection method can notice this abnormal traffic changes more easily. In **Fig. 9(a)**, with 15 minutes attack duration, any configuration of OTAT and STBAT achieves over 0.9 detection rate.

In the perspective of false positive rate, OTAT is affected by attack duration seriously, but STBAT is not affected that much by the attack duration. The main difference is traffic seasonality aware threshold adjustment. OTAT does not properly update threshold but suppress the threshold when continuous attack happens, and its false positive rate is greatly affected by the attack duration. After detecting an attack, OTAT keeps the threshold until the observed traffic becomes lower than the threshold. During OTAT suppress the threshold the legitimate traffic changes. Even though the continuous attack is done, the legitimate traffic

becomes bigger than the threshold and false positive cases happen. If the attack duration becomes longer and the duration of suppressing the threshold becomes longer, the ligitimate traffic changes dramatically and the probability of happening false positive cases increases. On the other hand, in STBAT, even though the attack happens continuously, we can estimate the amount of the legitimate traffic in every time window by using the traffic seasonality. That is, STBAT tries to keep track of the changes of legitimate traffic during long-lasting attacks, and the impact of attack duration is limited in STBAT.

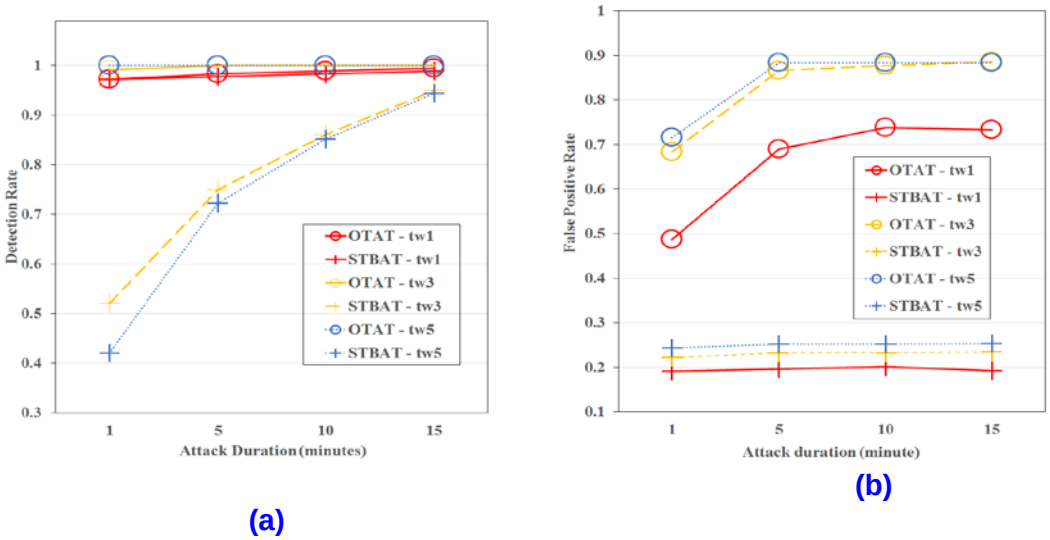



**Fig. 9.** Impact of attack duration on detection rate and false positive rate

Also in this evaluation we vary the time window size as 1 minute (tw1), 3 minutes (tw3) and 5 minutes (tw5). If we use bigger time window, we can diminish the requirements of computing resources on the gateway where the DoS detection system is deployed. However, with bigger time window, spontaneous peaks of attack traffic may mixed into legitimate traffic more easily. As we can see in **Fig. 9(a)**, the detection rate decreases when the time window size increases. Also, with bigger time windows, higher false positive rate is obtained as shown in **Fig. 9(b)**. In the case of OTAT, the time window size affects the false positive rate significantly, and with 3 minutes time windows it obtained near 0.9 false positive rates when attack duration exceeds 5 minutes. On the other hand, in the case of STBAT, when bigger time windows are used, the false positive rate increases up to around 0.25. Though STBAT achieves relatively low detection rate with bigger time windows such as 3 minutes and 5 minutes, the detection rate increases as an attack becomes more severe without any loss of false positive rate.

Accordingly, the tradeoff between the time window size and the performance of detection methods can be a design issue. If a sub network desires more secure property, STBAT with lower time window size is required and the network gateway needs more computing power. That is, if a sub network could not have powerful gateway but it still wants to detect very small attack peaks, it is recommended to have an aggregation point of multiple subnetworks where suitable computing power is supported.

## 5. Conclusion

In this paper, we proposed an effective source-side DoS detection method with a traffic seasonality aware adaptive threshold. The traffic volume aware adaptive threshold establishment helps the proposed method achieve high detection rate of subtle attack traffic. Moreover, the traffic seasonality aware threshold adjustment helps the proposed method decrease the false positive even though the legitimate traffic is highly fluctuated and subtle attack traffic is blended to the legitimate traffic. Especially, when the existence of attack traffic is detected before updating the threshold, the traffic seasonality helps to separate the legitimate traffic from the attack traffic and diminishes the side effect of attack traffic to the procedure of updating threshold. Extensive evaluation results with real datasets represent that the proposed method achieves up to 90% of detection rate and suppress the false positive down to 5%. In the future, we will study dedicated machine learning techniques for modeling the seasonality of traffic to adjust the threshold more properly, as well as for understanding traffic models of various IoT sub-domains in order to detect abnormal network activities.

## Acknowledgements

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (NRF-2017R1A2B4012559). This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2018-2016-0-00314) supervised by the IITP(Institute for Information & communications Technology Promotion).

## References

[1] Hakem Beitollahi and Geert Deconinck, "Analyzing well-known countermeasures against distributed denial of service attacks," *Computer Communications*, vol. 35, no. 11, pp. 1312-1332, June, 2012. Article (CrossRef Link).

[2] Hiroshi Tsunoda, Kohei Ohta, Atsunori Yamamoto, Nirwan Ansari, Yuji Waizumi and Yoshiaki Nemoto, "Detecting DRDoS attacks by a simple response packet confirmation mechanism," *Computer Communications*, vol. 31, no. 14, pp. 3299-3306, September, 2008. Article (CrossRef Link).

[3] Junjie Zhang, Xiapu Luo, Roberto Perdisci, Guofei Gu, Wenke Lee and Nick Feamster, "Boosting the scalability of botnet detection using adaptive traffic sampling," in *Proc. of the 6th ACM Symposium on Information, Computer and Communications Security*, pp. 124-134, March 22-24, 2011. Article (CrossRef Link).

[4] Ryoichi Kawahara, Tatsuya Mori, Noriaki Kamiyama, Shigeaki Harada and Shoichiro Asano, "A study on detecting network anomalies using sampled flow statistics," in *Proc. of 2007 International Symposium on Applications and the Internet Workshops*, pp. 81, January 15-19, 2007. Article (CrossRef Link).

[5] Zecheng He, Tianwei Zhang and Ruby B. Lee, "Machine Learning Based DDoS Attack Detection from Source Side in Cloud," in *Proc. of 2017 IEEE 4th International Conference on Cyber Security and Cloud Computing*, pp. 114-120, June 26-28, 2017. Article (CrossRef Link).

[6] N. Levy, D. Smith, and J. Schiel, "Operationalizing ISP cooperation during DDoS attacks," in *Proc. of North American Network Operators' Group Meeting 71*, October 2-4, 2017.

[7] Katerina Argyraki and David R. Cheriton, "Scalable Network-Layer Defense Against Internet Bandwidth-Flooding Attacks," *IEEE/ACM Transactions on Networking*, vol. 17, no. 4, pp. 1284-1297, August, 2009. Article (CrossRef Link).

[8] Jelena Mirkovic and Peter Reiher, "D-WARD: a source-end defense against flooding denial-of-service attacks," *IEEE transactions on Dependable and Secure Computing*, vol. 2, no. 3, pp. 216-232, September, 2005. Article (CrossRef Link).
[9] D. Senie and P. Ferguson, "Network ingress filtering: Defeating denial of service attacks which employ IP source address spoofing," *RFC 2267*, January, 1998. Article (CrossRef Link).
[10] Min Suk Kang, "Revisiting Source-end DDoS Filtering in the New Age," in *the 45th Asia-Pacific Advanced Network Meeting*, March 25-29, 2018.
[11] Jelena Mirkovic, Gregory Prier, and Peter Reiher, "Attacking DDoS at the source," in *Proc. of the 10th IEEE International Conference on Network Protocols*, pp. 312-321, November 12-15, 2002. Article (CrossRef Link).
[12] C. Morrow and R. Dobbins, "DDoS Open Threat Signaling (DOTS) Working Group Operational Requirements," *Active Internet-Draft*, Last updated 2018-11-23
[13] Dimitris Gavrilis and Evangelos Dermatas, "Real-time detection of distributed denial-of-service attacks using RBF networks and statistical features," *Computer Networks*, vol. 48, no. 2, pp. 235-245, June, 2005. Article (CrossRef Link).
[14] Gavrilis Dimitris, Tsoulos Ioannis and Dermatas Evangelos, "Feature selection for robust detection of distributed denial-of-service attacks using genetic algorithms," in *Proc. of Hellenic Conference on Artificial Intelligence (SETN 2004)*, pp. 276-281, 2004. Article (CrossRef Link).
[15] Sinh-Ngoc Nguyen, Jintae Choi and Kyungbaek Kim, "Suspicious traffic detection based on edge gateway sampling method," in *Proc. of the 19th Asia-Pacific Network Operations and Management Symposium*, pp. 243-246, September 27-29, 2017. Article (CrossRef Link).
[16] Sinh-Ngoc Nguyen, Van-Quyet Nguyen, Giang-Truong Nguyen, JeongNyeo Kim and Kyungbaek Kim, "Source-Side Detection of DRDoS Attack Request with Traffic-Aware Adaptive Threshold," *IEICE Transactions on Information and Systems*, vol. E101.D, no. 6, pp. 1686-1690, June, 2018. Article (CrossRef Link).
[17] Simple exponential smoothing. Article (CrossRef Link).
[18] About DNS-STATS:Hedgehog. Article (CrossRef Link).
[19] Domain Name System Operations Analysis and Research Center. Article (CrossRef Link).
[20] Enzo Baccarelli, Nicola Cordeschi, Alessandro Mei, Massimo Panella, Mohammad Shojafar and Julinda Stefa, "Energy-efficient dynamic traffic offloading and reconfiguration of networked data centers for big data stream mobile computing: review, challenges, and a case study," *IEEE Network*, vol. 30, no. 2, pp. 54-61, March, 2016. Article (CrossRef Link).
[21] Peng Xiao, Wenyu Qu, Heng Qi and Zhiyang Li, "Detecting DDoS attacks against data center with correlation analysis," *Computer Communications*, vol. 67, pp. 66-74, August, 2015. Article (CrossRef Link).
[22] Quamar Niyaz, Weiqing Sun and Ahmad Y. Javaid, "A deep learning based DDoS detection system in software-defined networking (SDN)," *EAI Endorsed Transactions on Security and Safety*, vol. 4, no. 12, December, 2017. Article (CrossRef Link).
[23] Rup Kumar Deka, Kausthav Pratim Kalita, D.K. Bhattacharya and Jugal K. Kalita, "Network defense: Approaches, methods and techniques," *Journal of Network and Computer Applications*, vol. 57, pp. 71-84, November, 2015. Article (CrossRef Link).
[24] Adrian Lara, Anisha Kolasani, and Byrav Ramamurthy, "Network innovation using openflow: A survey," *IEEE communications surveys & tutorials*, vol. 16, no. 1, pp. 493-512, 2014. Article (CrossRef Link).
[25] Chu YuHunag, Tseng MinChi, Chen YaoTing, Chou YuChieh and Chen YanRen, "A novel design for future on-demand service and security," in *Proc. of 2010 12th IEEE International Conference on Communication Technology*, pp. 385-388, November 11-14, 2010. Article (CrossRef Link).
[26] F. J. Ryba, M. Orlinski, M. Wahlisch, C. Rossow and T. C. Schmidt, "Amplification and DRDoS Attack Defense--A Survey and New Perspectives," *arXiv:1505.07892*, 2015.

**Giang-Truong Nguyen** received the B.Sc degree in information technology from the Hanoi University of Science and Technology (HUST), Ha Noi, Viet Nam, in 2015. He is currently pursuing the M.Sc degree in computer science from the School of Electronics and Computer Engineering at the Chonnam National University, South Korea. His research interests are in Social Network Analysis and User Behavior Analytics.

**Van-Quyet Nguyen** received the B.Sc degree in information technology from the Hung Yen University of Technology and Education, Hung Yen, Viet Nam, in 2009. He received the M.Sc in computer and information science from the Hanoi University of Science and Technology (HUST), Ha Noi, Viet Nam, in 2013. He is currently pursuing the Ph.D degree in computer science from the School of Electronics and Computer Engineering at the Chonnam National University, South Korea. His research interests are in Big Graph Analytics, Big Data Platforms, Social Network Analysis, and Internet of Things. He is also a lecturer at the Hung Yen University of Technology and Education.

**Sinh-Ngoc Nguyen** received the B.Sc degree in computer engineering from the University of Information Technology at Viet Nam National University Ho Chi Minh City, Viet Nam, in 2015. He received the M.Sc degree in computer science from the School of Electronics and Computer Engineering at the Chonnam National University, South Korea, in 2018. Currently, he is a software developer at Xisom company which provides the software solutions for automation system. His research interests are SDN, Network Security, Big Data Analysis and Big Data Platforms.

**Kyungbaek Kim** received the BS, MS, and PhD degrees in electrical engineering and computer science from Korea Advanced Institute of Science and Technology (KAIST), Korea, in 1999, 2001, and 2007, respectively. He is currently an Assosiate Professor in the School of Electronics and Computer Engineering at the Chonnam National University, South Korea. Previously, he was a Postdoctoral Researcher in the Department of Computer Sciences, University of California Irvine, CA, USA. His research interests are in the design and implementations of distributed systems including peer-to-peer systems, Grid/Clouding systems, Big Data platform and social networking systems. He is a member of USENIX, IEEE, IEICE and KSII.