Using the Cumulative Sum Algorithm Against Distributed Denial of Service Attacks in Internet of Things

Pheeha Machaka[1(✉)], Andre McDonald[1], Fulufhelo Nelwamondo[1], and Antoine Bagula[2]

[1] Council for Scientific and Industrial Research, Modelling and Digital Science, Meiring Naude Rd, Pretoria 0184, South Africa
{PMachaka, AMcdonald, FNelwamondo}@csir.co.za
[2] University of the Western Cape, Robert Sobukwe Road, Bellville 7535, South Africa
BBagula@uwc.ac.za

Abstract. The paper presents the threats that are present in Internet of Things (IoT) systems and how they can be used to perpetuate a large scale DDoS attack. The paper investigates how the Cumulative Sum (CUSUM) algorithm can be used to detect a DDoS attack originating from an IoT system, and how the performance of the algorithm is affected by its tuning parameters and various network attack intensities. The performance of the algorithm is measured against the trade-off between the algorithm's detection rate, false alarm and detection delay. The performance results are analysed and discussed and avenues for future work are provided.

Keywords: Anomaly detection · Internet of things · Change detection Distributed denial of service · TCP SYN flooding · Cumulative sum · Intrusion detection

1 Introduction

The recent advances in Information Communications Technology (ICT) have led to a new era called the Internet of Things (IoT). In this paradigm, many of the objects (or things) that surround our living environment will be connected to the Internet or another network in one form or another. The services and applications provided by these technologies may include smart electricity meter reading, intelligent transportation, stock exchanges monitoring and health monitoring [1].

IoT allows everyday use objects like the smartphones, smart-TV, smart-fridge and many other smart devices to be connected to the Internet. This trend will keep on growing as more objects gain the means and capacity to directly interface with the Internet. The use of these services and devices will result in enormous amounts of data being generated by these inter-connected devices. This data needs to be stored, processed and presented in a seamless, efficient, and easily interpretable form [1]. Of most importance is the security and privacy of the services provided by these technologies

C.V. Phan and A. Vangalur (Eds.): ICCASA 2015, LNICST 165, pp. 62–72, 2016.
DOI: 10.1007/978-3-319-29236-6_7

that ensures the confidentiality, integrity and authenticity of the data in IoT. This is because each device and service is susceptible to abuse by attackers. Therefore the escalated use of IoT raises several security vulnerabilities [2].

The vulnerabilities and concerns that are present in IoT are due to the characteristics that make up the IoT architecture. The one concern is the increased population of objects; this presents an opportunity for attackers to use them as an army of zombies to carry out a large scale attack [1]. Another concern is the ubiquity, mobility and interoperability of IoT systems. The ubiquity and physical distribution of IoT devices provides a window of opportunity for attackers to gain physical access and get closer to the target system [2]. The IoT system's increased mobility and interoperability intensifies the threat of IoT systems such that the attacker may gain access to the system and institute infected devices into the system in order to further jeopardise the system and evade detection of a large scale attack [1].

The IoT system presents an opportunity for attackers to launch a large scale Distributed Denial of Service (DDoS). A DDoS attack is a malicious attempt by an attacker to disrupt the online services of a service provider to make it unavailable to its legitimate users. This may lead to disgruntled service consumers and major financial losses; it may also lead to losses in an organization's intellectual property which in turn affects the long term competitiveness of businesses and governments in industrial and military espionage incidents [1]. It is therefore important that organizations and governments deploy methods and techniques that will help them to accurately and reliably detect the onset and occurrence of the DDoS attacks.

This paper investigates the performance of the CUSUM algorithm against a DDoS attack. The paper also investigates the trade-off between the algorithm's detection rate, false alarm and detection delay. The paper seeks to further investigate how the performance of the algorithm is affected by the tuning parameters and how various network attack intensity affect its performance.

The remainder of the paper is organised as follows: In Sect. 2 a brief review of a DDoS flooding attack and related work is provided. Section 3 discusses the research and experiment design while Sects. 4 and 5 presents and discusses the results obtained from the experiments.

2 Background and Related Work

An attacker uses DDoS attacks in order to prevent legitimate users from accessing the service of a provider. The attacker does this through the use of an attack that streams multiple illegitimate requests to the victim, e.g. a High-Rate Flooding (HRF) attack. There have been various classifications of DDoS attacks in the literature [3–8], however the focus of this paper will be on the malicious and widely used TCP SYN flooding attack.

2.1 TCP SYN Flooding Attack

A TCP SYN flooding attack is an example of a network layer flooding attack, and it is one of the most common and powerful flooding methods. It exploits the vulnerabilities

of the TCP three-way handshake. In a normal TCP connection, the client initiates the connection by sending a SYN packet to the server, as a way of requesting a connection. Upon receiving the connection request, the server will open a connection session and respond with a SYN_ACK packet; by doing this the server stores details of the requested TCP connection in the memory stack and allocates resources to this open session. The connection remains in a half-open state, i.e. the SYN_RECVD state. To complete the three-way handshake with the server, the client will need to confirm the connection and respond with an ACK packet. The server will then check the memory stack for an existing connection request, and the TCP connection will be moved from the SYN_RECVD state to ESTABLISHED state. If there is no ACK packet sent within a specific period of time, the connection will timeout and therefore releasing the allocated resources [5, 8, 9].

In a TCP SYN flooding attack, the attacker streams large volumes of SYN packets towards the victim server. A vulnerable IoT system can be used for this purpose. These packets normally contain spoofed IP addresses, i.e. IP addresses that are non-existent or are not utilised. TCP SYN floods can also be launched using compromised machines with legitimate IP addresses, however the machines need to be configured in such a way that it does not respond or acknowledge a SYN_ACK packet from the victim server. In this way the server will not receive any ACK packet from the clients for the 'half-open' connection request. During the high rate flooding attack, and for a period of time, the server will maintain a large volume of incomplete three-way handshake and allocates resource towards the fictitious connection requests. The server will gather more fictitious requests and eventually exhaust its resources. This will prevent new requests, including legitimate client requests, from being further processed by the server [5, 8, 9].

2.2 Anomaly and Change Detection Algorithms

In the event of a DDoS attack, abrupt changes in observed network traffic can be observed. Similarly, an abrupt change in statistical properties of detection features can be observed. Thus, the problem of anomaly detection can be constructed as change point detection problem [10, 11]. The aim of change detection techniques are to help detect a change in statistical properties of observed network traffic with minimal detection delay and false positive rate [12]. The approach first starts by applying filter to the traffic data by desired features and arraigning the data into a time series data. For change detection, if there was a DDoS attack at time λ, the time series will show a significant statistical change around or at a time greater than λ [13].

Detecting changes in statistical properties of observed network traffic has been studied extensively and applied in various fields like image processing, network traffic and financial analysis. There are a number of techniques that are used for change detection and amongst them the most common technique used for the detection of DDoS attacks is the Cumulative Sum (CUSUM) algorithm [14].

2.3 Cumulative Sum (CUSUM) Algorithm

Several variants of the CUSUM algorithm were first introduced by Page in [15]. The CUSUM algorithm is based on hypothesis testing and was developed for independent and identically distributed random variables $\{y_i\}$. In the approach, an abrupt change occurring at any time can be modeled using two hypotheses, θ_0 and θ_1. The first hypothesis θ_1 represents the statistical distribution before the abrupt change occurring; and the second hypothesis θ_2 represents the statistical distribution after the abrupt change has occurred. The test for signaling a change is based on the log-likelihood ratio S_n.

$$S_n = \sum s_i$$

Where,

$$s_i = \ln \frac{P_{\theta_1(y_i)}}{P_{\theta_2(y_i)}}$$

According to Siris et al. [9]. the typical behavior of the log-likelihood ratio S_n includes a negative divergence before an abrupt change and a positive divergence after the change. Therefore, the relevant information for detecting a change lies in the difference between the value of the log-likelihood ratio and its current minimum value [12]. The alarm condition for the CUSUM algorithm takes the form:

If $g_n \geq h$ (h is a threshold parameter) then signal alarm at time n;

$$\text{where } g_n = S_n - m_n \tag{1}$$

and

$$m_n = \min_{1 \leq j \leq n} S_j. \tag{2}$$

In the above equations, it is assumed that $\{y_i\}$ are independent Gaussian random variables with known variance σ^2 and mean μ_0 and μ_1 represents the mean before and after the abrupt change, respectively. Accordingly, $\theta_0 = N(\mu_0, \sigma^2)$ and $\theta_1 = N(\mu_1, \sigma^2)$. Following an application of various calculations, Basseville et al. [12] implemented the following CUSUM algorithm:

$$g_n = \left[g_{n-1} + \frac{\mu_0 - \mu_1}{\sigma^2} \left(y_n - \frac{\mu_1 + \mu_0}{2} \right) \right]^+ \tag{3}$$

The above algorithm was adapted and applied to the problem of detecting SYN flooding attacks. This algorithm was applied as follows [9]:

$$\tilde{x}_n = x_n - \bar{\mu}_{n-1} \tag{4}$$

where x_n represents the number of SYN packets in the n-th time interval, and $\bar{\mu}_n$ represents the estimated mean rate at time n. The estimates mean rate is computed using an exponentially weighted moving average as follows:

$$\bar{\mu}_n = \beta\bar{\mu}_{n-1} + (1-\beta)x_n \tag{5}$$

where β is the exponentially weighted moving average (EWMA) factor.

The mean value of $\tilde{x}_n$ prior to a change is zero, therefore the mean in (3) is $\mu_0 = 0$. The mean traffic rate after a change cannot be known in advance. It can therefore be estimated with $\alpha\ \bar{\mu}_n$, were α is an amplitude percentage parameter. The parameter equates to the most likely percentage increase of the mean rate after an attack has occurred. For purposes of detecting SYN flood attacks, the algorithm in (3) has been adapted to:

$$g_n = \left[g_{n-1} + \frac{\alpha\bar{\mu}_{n-1}}{\sigma^2}\left(x_n - \bar{\mu}_{n-1} - \frac{\alpha\bar{\mu}_{n-1}}{2}\right)\right]^+ \tag{6}$$

In the CUSUM algorithm, the tuning parameters are the amplitude factor, α, the Weighting factor, β, and the CUSUM algorithm threshold, h.

2.4 Related Work

The CUSUM technique has been applied to various problems including DDoS detection. It calculates the cumulative sum of difference between actual and expected values of a sequence, the CUSUM value. This value is compared to a threshold value (an upper bound). A CUSUM value greater than the upper bound threshold indicates a change in statistical properties in the observed network traffic time series values.

There are a number of variations of the CUSUM technique, and Tartakovsky et al. [10] proposed fully-sequential and batch-sequential algorithms. They are both non-parametric variations of the CUSUM techniques adapted to detecting changes in multiple bins. The algorithms were found to be self-learning, which enables them to adapt to various network loads and usage patterns. They also allow for the detection of attacks with a small average delay for a given false-alarm rate and they are computationally feasible and thus can be implemented online.

Bo et al. [16] also used an algorithm which is a variation of the CUSUM technique to enable quick detection of worm attack incidents. In their experiments they observed the computer's degree of connection to estimate the CUSUM score. It was concluded that the algorithm could detect new attacks fast and effectively.

There has been various combination of the CUSUM technique with other detection techniques. For example, Dainotti et al. [17] used a combination of the adaptive threshold, Continuous Wavelet Transform (CWT) and the CUSUM technique to detect volume based attacks. In this proposed techniques, two detection engines were used. An anomaly will be detected by the first stage detection engine will detect an anomaly, and an alarm is sent to the second stage detection engine (based on CWT) which refines the detection in order to avoid high number of false alarms.

Siris et al. [9] proposed and investigated a change point detection algorithm, which is also based on the CUSUM technique. The algorithm revealed robust performance over various attack types, it was computationally feasible and not computationally expensive. Wang et al. [18] also proposed an algorithm which is a variation of the

CUSUM technique on an application for detecting DDoS attacks. Protocol behaviors of TCP SYN – FIN (RST) pairs where used to make detections. The experiment results revealed that the algorithm had a low detection delays and high detection accuracy.

3 The Research Design

The experiments were conducted using actual network traffic data from the MIT Lincoln Laboratory, the DARPA intrusion detection dataset. The data contains trace data taken during a day of network activity. The experiment considered trace data where there was significant traffic activity. Therefore trace data between the times 08h00–19h00 were considered, and thus an 11 h period of real network packets was used for experiments.

To allow for investigations of the algorithm's performance across different types of attack characteristics, the attacks were generated synthetically to simulate an IoT system. They were generated as a homogenous Poisson process with independent and exponentially distributed delays between packet arrivals. The synthetically generated attack was designed to last for 300 s (5 min) over 30 time intervals (using a 10 s time interval). To consider all possible attack combinations within the 11 h network packet trace, each 5 min window was injected with attack data in separate runs of the experiments.

In these experiments we consider and simulate two types of attack characteristics: high intensity and low intensity attacks. Low intensity attacks are those attacks whose intensity increases gradually. In these experiments we considered the case of a low intensity attack to be an attack that, within the 5 min attack interval, has its mean amplitude to be 50 % above the actual attack free traffic's mean rate. High intensity attack are those attacks whose intensity increases abruptly and reach a peak amplitude within one time interval. High intensity attacks were considered to be attacks that are 250 % higher than the peak rate within the 5 min attack interval.

4 Results and Discussions

These experiments were investigating the performance of the CUSUM algorithm for both low and high intensity attacks, testing the following: (1) the effect, on detection-rate, of the amplitude factor α, tuning parameter; (2) the effect, on detection rate, of the weighting factor β, tuning parameter; (3) the trade-off between detection rate and the false positive rate; (4) the trade-off between the detection rate and detection delay. The result and discussion from the experiments are expanded in the sub-sections that follow.

4.1 The Effect of the Amplitude Factor (α)

In this section the effect of the amplitude factor (α) on the detection rate and the false positive rate is investigated. In this part of the experiments, the value of the weighting

factor was held constant β = 0.8; the CUSUM amplitude factor h = 3; while the amplitude factor was varied between [0.05; 1.0].

The Fig. 1 depicts the results of the experiments. From the Fig. 1(a) it can be observed that for low rate attacks, the detection algorithm yields a detection rate between 0 % – 81 % and a false positive between 0 % – 7 % for values of $0 < \alpha \leq 0.5$.

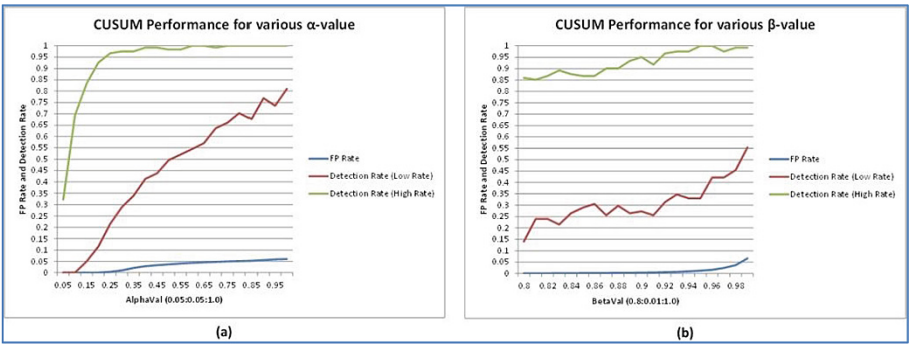

Fig. 1. Detection rate and False Alarm rate for (a) varied amplitude factor value (α); and (b) varied Weighting factor (β), for both Low rate attacks and High rate attacks.

From the Fig. 1(a), it can be observed that for high rate attacks and values of $0 < \alpha \leq 0.95$, the detection algorithm yields a 32 % – 100 % detection rate while having a false positive rate between 0 % – 6 %. The above experiment signifies that when the value of the amplitude factor increases, the values of the detection rate and the false positive rate also increases.

4.2 The Effect of the Weighting Factor (β)

In this section the effect of the value of the Weighting factor (β) on the detection rate and the false positive rate is investigated. In this part of the experiments, the value of the amplitude factor was made constant α = 0.5; the CUSUM amplitude factor h = 3; while the value of the Weighting factor (β) was varied between [0.80; 1.0].

From the Fig. 1(b) it can be observed that the performance of the detection algorithm was poor. For low rate attacks, the algorithm had a detection rate between 14 %-56 %; while the false positive rate was between 0 % - 7 %. For high rate attacks, shown by Fig. 1(b), the detection rate was between 85 % and 100 %; In this experiment, the CUSUM algorithm reached a 100 % detection rate for values of $\beta \geq 0.95$. However, the higher detection rate was accompanied by an increased false positive rate. Therefore there is a trade-off between a higher detection rate and an increased false positive rate.

4.3 Trade-off Between False Positive Rate and Detection Rate

In this set of experiments we were investigating the trade-off between false positive rate and detection rate. The values for the tuning parameters were as follows: the amplitude factor $\alpha = 0.5$; the Weighting factor $\beta = 0.98$. The CUSUM amplitude factor h was varied from 1 to 10.

Figure 2 displays the Receiver operating curves for the experiments, where each point corresponds to a different value of h. Good operating points on the graph are those points that are closer to the upper-left corner of the graph. Figure 2(a) shows results for the experiments simulating low rate attacks. In the case of low rate attacks, an increase in the algorithm's detection accuracy was accompanied by a sharp increase in the false alarm rate. Therefore higher detection accuracy will also result in a higher false alarm rate. This is a performance that is not desired in a detection algorithm. Therefore the CUSUM algorithm did not perform very well for cases of low rate attacks.

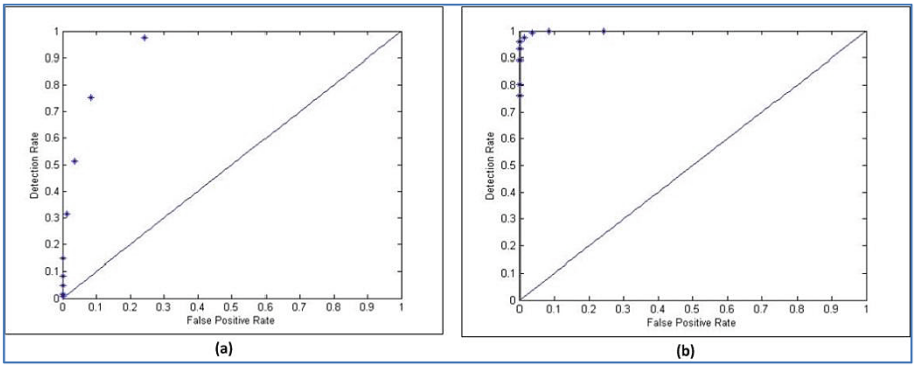

Fig. 2. Receiver Operating Curves for trade-off between FP Rate and Detection Rate for (a) low rate attacks and (b) high rate attacks.

Figure 2(b) depicts the performance of the CUSUM algorithm in the case of high rate attacks simulation. Most of the operating points are closer to the upper-left corner of the graph. This is also indicative that for a higher detection rate there is a slight increase in the false alarm rate. This is an improved algorithm performance when compared with the low rate attack simulations.

4.4 Trade-off Between Detection Rate and Detection Delay

In the next set of experiments we further analyzed the trade-off between detection rate and detection delay. The results are shown in Fig. 3 below. Detection delay in this case is the average time taken by the algorithm to successfully detect an attack, from the onset of that attack. Each point corresponds to a pair of detection rate and average detection delay. The values for the tuning parameters were as follows: the amplitude factor $\alpha = 0.5$; the Weighting factor $\beta = 0.98$. The value of the amplitude factor h was varied from 1-10.

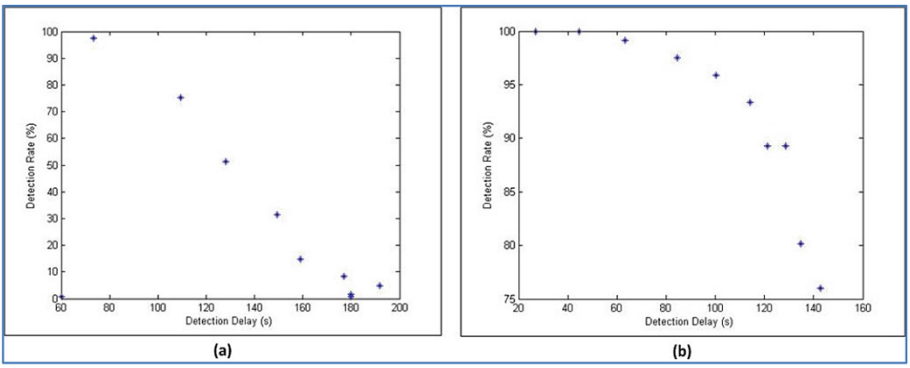

Fig. 3. Graph displaying the trade-off between Detection Rate and average Detection Delay for (a) low rate attacks and (b) high rate attacks.

Figure 3(a) depicts the trade-off between detection rate and average detection delay performance of the CUSUM algorithm for low rate attack simulations. From the graph it can be observed that as the detection rate decreases, the accompanying detection delay increases. From the simulation with low rate attacks, the CUSUM failed to reach a 100 % detection rate, but the best average detection delay was just below 80 s (73.3 s).

Figure 3(b) displays results for the simulations with high rate attacks. The algorithm had an improved performance for high rate attacks. For a 100 % detection rate the average detection delays was at 26.94 s and 44.71 s for varying h-values. It can also be observed that for lower detection rate performance the average detection delay also increases.

5 Conclusions

This paper described, analyzed and discussed how the CUSUM algorithm can be used for detecting DDoS attacks in an IoT system. The simulation experiments investigated how the performance of the algorithm is affected by the tuning parameters (α, β and h). These were efforts to find optimal parameter tuning for best CUSUM algorithm performance. It also investigated the trade-off between detection rate and false positive rate, as well as detection rate and average detection delay. Furthermore, the experiments were conducted on real network traffic data with synthetically generated attack data with two levels of attack intensity (i.e. low rate and high rate attacks).

In these experiments it was found that optimal CUSUM parameter tuning for this network traffic was: $\alpha = 0.5$, $\beta = 0.98$ and $h = 3$. Furthermore, it was found that the CUSUM algorithm performs well for high rate attacks, however its performance subsides for low rate attacks. This further confirms the findings by the authors in [9].

Ongoing research work will include performance comparison of the CUSUM with other anomaly detection algorithm, similar to the work of authors in [19–23]. Another avenue for future work is the development of change detection algorithms that perform well under various attack characteristics and intensity.

References

1. Uckelmann, D., Harrison, M., Michahelles, F.: An architectural approach towards the future internet of things. In: Architecting the Internet of Things (2011)
2. Weber, R.H.: Internet of things-new security and privacy challenges. Comput. Law Secur. Rev. **26**(1), 23–30 (2010)
3. Douligeris, C., Mitrokotsa, A.: DDoS attacks and defense mechanisms: classification and state-of-the-art. Comput. Netw. **44**(5), 643–666 (2004)
4. Mirkovic, J., Reiher, P.: A taxonomy of DDoS attack and DDoS defense mechanisms. ACM SIGCOMM Comput. Commun. Rev. **34**(2), 39–53 (2004)
5. Zargar, S.T., Joshi, J., Tipper, D.: A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks. IEEE Commun. Surv. Tutorials **15**(4), 2046–2069 (2013)
6. Lazarevic, A., Kumar, V., Srivastava, J.: Intrusion detection: a survey. In: Managing Cyber Threats (2005)
7. Bhattacharyya, D.K., Kalita, J.K.: Network Anomaly Detection A Machine Learning Perspective. CRC Press, Boca Raton (2013)
8. Zhou, C.V., Leckie, C., Karunasekera, S.: A survey of coordinated attacks and collaborative intrusion detection. Comput. Secur. **29**(1), 124–140 (2010)
9. Siris, V.A., Papagalou, F.: Application of anomaly detection algorithms for detecting SYN flooding attacks. Comput. Commun. **29**(9), 1433–1442 (2006)
10. Tartakovsky, A.G., Rozovskii, B.L., Blazek, R.B., Kim, H.: A novel approach to detection of intrusions in computer networks via adaptive sequential and batch-sequential change-point detection methods. IEEE Trans. Signal Process. **54**(9), 3372–3382 (2006)
11. Tartakovsky, A.G., Polunchenko, A.S., Sokolov, G.: Efficient computer network anomaly detection by change point detection methods. IEEE J. Sel. Top. Sign. Process. **7**(1), 4–11 (2013)
12. Basseville, M., Nikiforov, I.V.: Detection of Abrupt Changes: Theory and Application, vol. 104. Prentice Hall, Englewood Cliffs (1993)
13. Poor, H.V., Hadjiliadis, O.: Quickest Detection, vol. 40. Cambridge University Press, New York (2009)
14. Carl, G., Kesidis, G., Brooks, R.R., Rai, S.: Denial-of-service attack-detection techniques. IEEE Internet Comput. **10**(1), 82–89 (2006)
15. Page, E.: Continuous inspection schemes. Biometrika **41**, 100–115 (1954)
16. Bo, C., Fang, B., Yun, X.: A new approach for early detection of internet worms based on connection degree. In: Proceedings of 2005 International Conference on Machine Learning and Cybernetics (2005)
17. Dainotti, A., Pescapé, A., Ventre, G.: Wavelet-based detection of DoS attacks. In: Global Telecommunications Conference, GLOBECOM 2006. IEEE (2006)
18. Wang, H., Zhang, D., Shin, K.G.: Detecting SYN flooding attacks. In: Proceedings of Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies INFOCOM 2002, IEEE (2002). DOI: 10.1109/INFCOM.2002.1019404
19. Machaka, P., Bagula, A., De Wet, N.: A highly scalable monitoring tool for wi-fi networks. In: 2012 IEEE 1st International Symposium on Wireless Systems (IDAACS-SWS) (2012)
20. Machaka, P., Bagula, A.: An investigation of scalable anomaly detection techniques for a large network of wi-fi hotspots. In: Jung, J.J., Badica, C., Kiss, A. (eds.) INFOSCALE 2014. LNICST, vol. 139. Springer, Heidelberg (2014). doi:10.1007/978-3-319-16868-5

21. Tran, D.Q., Nguyen, M.H.: Drought monitoring: a performance investigation of three machine learning techniques. In: Vinh, P.C., Alagar, V., Vassev, E., Khare, A. (eds.) ICCASA 2013. LNICST, vol. 128, pp. 47–56. Springer, Heidelberg (2014)
22. Bagula, A., Machaka, P., Mabande, T.: Monitoring of a large Wi-Fi hotspots network: performance investigation of soft computing techniques. In: Hart, E., Timmis, J., Mitchell, P., Nakamo, T., Dabiri, F. (eds.) BIONETICS 2011. LNICST, vol. 103, pp. 155–162. Springer, Heidelberg (2012)
23. Bagula, A., Machaka, P.: Preemptive performance monitoring of a large network of Wi-Fi Hotspots: an artificial immune system. In: Masip-Bruin, X., Verchere, D., Tsaoussidis, V., Yannuzzi, M. (eds.) WWIC 2011. LNCS, vol. 6649, pp. 494–504. Springer, Heidelberg (2011)