Discriminating DDoS Attack Traffic from Flash Crowd through Packet Arrival Patterns

Theerasak Thapngam, Shui Yu, Wanlei Zhou and Gleb Beliakov
School of Information Technology
Deakin University
Burwood, VIC 3125, Australia
Email: {tthap, syu, wanlei, gleb}@deakin.edu.au

Abstract—Current **DDoS attacks are carried out by attack tools, worms and botnets using different packet-transmission strategies and various forms of attack packets to beat defense systems. These problems lead to defense systems requiring various detection methods in order to identify attacks. Moreover, DDoS attacks can mix their traffics during flash crowds. By doing this, the complex defense system cannot detect the attack traffic in time. In this paper, we propose a behavior based detection that can discriminate DDoS attack traffic from traffic generated by real users. By using Pearson's correlation coefficient, our comparable detection methods can extract the repeatable features of the packet arrivals. The extensive simulations were tested for the accuracy of detection. We then performed experiments with several datasets and our results affirm that the proposed method can differentiate traffic of an attack source from legitimate traffic with a quick response. We also discuss approaches to improve our proposed methods at the conclusion of this paper.**

Keywords-component; DDoS attacks; correlation coefficient; anomaly detection; traffic patterns;

I. INTRODUCTION

Current DDoS attacks remain a high threat to IT security on the Internet. The attacks can be carried out by attack tools [1], worms [2], and botnets [3] with attack variants of packet transmission such as TCP/SYN, UDP and HTTP request floods [4]. These sources of DDoS attack are powerful and can overwhelm any online host and server. Moreover, one of the biggest challenges for DDoS attack detection is *flash-crowd attack*. Flash-crowd attack [5][6][7] is the phenomenon of a high volume of illegitimate packets from attack sources. The attack traffic is viewed the same as legitimate users' traffics (called *flash crowd*). Attack sources pretend to be real users and pump a large volume of request packets that flood the target victim. In this case, the defense/detection system could be beaten and the server has difficulty surviving the attack which causes it to crush or degrade the service.

Statistical-based defense systems [8][9][10] against DDoS attacks rely on header information from IP packets such as IP address, time-to-live (TTL), protocol type (port number), etc. The detection can discriminate "normal" traffic from "abnormal" traffic which is more likely to be an attack. However, some botnets, e.g. Mydoom [2] can bypass detection approaches through the victim. This is because the approaches consider the Transport layer and/or Network layer. Therefore, the botnets which generate similar legitimate HTTP packets can avoid detection. Even though the attacking HTTP traffic is aggregated, they still look like flash crowd.

Heuristic-based defense systems [4][11] against DDoS attack rely on a threshold. Each approach may need to calculate its own threshold to judge the current observing traffic. The drawback of heuristic detection approaches is their inability to consider legitimate traffic mixed with attacking traffic. Hence, packets from legitimate users may be blocked or eliminated during attack incidents occur.

Since attack sources have been programmed and worked according to their attack functions, pattern detection based on their behaviors is possible. The worms work as an automatic program which can be differentiated from human users. The botnets and DDoS attack tools work as a semi-automatic program after an attacker issues the attack command based on C&C fashion. Hence, these attack sources could repeatedly generate attack packets with different transmission abilities. These anomaly behaviors could be predictable and explainable in pattern styles. In contrast, the arrival rate based on human users, including a proxy server seems to constitute the nonpatternable (random) cases.

In this paper, we propose a solution to detect the pattern behavior of traffic sources by observing packet arrivals. This proposed technique is an effective method to discriminate packets among DDoS attack sources and real users including proxies. We will use the packet arrival rate as information to differentiate attack-source traffic from user traffic. We have provided more details in Section 2. Since we can measure the degree of pattern behavior, we can push the right actions to the right packets. The packets from the attack sources must be eliminated, but the user packets must get through the server.

The contributions of the paper are listed as follows:

- *Fast detection*: The detection system must be able to detect the DDoS attacks in time.
- *Reliability*: The detection system must not cause false positive and false negative in results.
- *Feasibility*: The detection system must be able to implement in real-world cases based on current Internet technology.
- *Real-time implementation*: The detection system must be able to respond as soon as the flash-crowd traffic arrives at the server.
- *Flexibility*: The detection system must be able to detect any form of attack packets such as malformed IP, TCP, UDP, ICMP, Application-based floods, etc.

The rest of the paper is organized as follows. Section 2 reviews the background of our research. Section 3 states the problem and defines the methods to solve the problem. Each method will be discussed detailing the threshold and variables in Section 4. The next Section uses the adjusted threshold and variables to experiment with the real datasets. In the final Section, we provide the summary and talk about the direction of our research in the future.

II. Background

Our proposed approach to discriminate DDoS attack traffic from user traffic is to observe the packet transmission rate. An individual host may require access to a server by sending a request. The request packets can be, for example, TCP/SYN, or HTTP requests, etc. Hence, the request packet transmission can be observed using the degree of automation, as we know that attack sources work following the instructions from the programmer and have a very high degree of automation to work after instructions are issued. When the attack sources perform a DDoS attack on the victim, their transmission rate appears to be predictable and itself becomes a pattern in a short period of time. However, Internet users have a limit for the response from the outcome after the first request. For example, after a webpage has been shown, the user may take time to skim and respond, for example, clicking on a link. In other words, human users unpredictably create request packets at any period of time. Hence, we can test the pattern of packet transmission by using some mathematical models or statistical analysis.

A. Predictable and Nonpredictable Rates

As we know, attack rates depend on the characteristics of packet transmission. In the victim-end point, the attack packets that are received can be observed as an arrival rate. This attack behavior can be divided into two main types as follows:

1) Predictable attack rate: The attack agents (botnets) send out the attack packets in a predictable sense to the victim. For instance, if we have enough data of a packet arrival at the time interval, we would know what is going to occur at the next time interval. This is important behavior of a botnet which is an automatic program. The program follows the instructions from the (malicious) programmer. The botnet program usually repeats packet tranmission until other commands are issued. There are various arrival rates (attack rates) and they can be classified as follows:

a) Constant rate attack [6][12] can be considered a stable attack rate. The attack agent (botnet) may use a constant attack rate that may be considered from the available bandwidth, the performance of a computer, and so forth. With a low bandwidth rate, the attack can fly under the radar and get through the defense system. Therefore, this attack can disturb and/or reduce the quality of services until a denial of services occurs that depends on the aggregate rate at the victim site. In cases of DDoS attack, the attack agents may continue sending the attack packets to the victim with maximum available bandwidth and full ability for transmission. This could destroy the victim's service. When a large number of agents flood a huge number of attack packets simultaneously, the vulnerable victim will be overwhelmed and unable to serve legitimate client requests. In a worst case scenario, the victim's servers can completely crash.

b) Increasing rate attack [6][12] can be considered a linear or an exponential attack rate and is also known as an abrupt rate attack. The attack agent may increase its packet transmission rate gradually or dramatically. As a result, the victim's resources are either slowly or rapidly exhausted. A slowly increasing attack rate can delay sensory detection of an attack. The attack agent, however, may increase the attack rate to maximum or decrease its attack rate at a later stage.

c) Periodical rate attack [6][13][14] generates a predictable attack rate. The attack agents may not continue the same attack rates, but may repeat transmission behavior of attack packets as a regular pattern. A periodical rate attack is also defined in Pulsing DoS attack which considers period of attack (T), length of the peak (L), and magnitude of the peak (R) [7][15].

2) Nonpredictable attack rate: The variable rate attack (or fluctuating rate attack) [12] is varying the transmission rate of attack packets to avoid detection and response. To generate an unpredictable attack rate, the attack agents may randomize the transmission rate and the attack delay time for the attack packets. The attack could be generated in continuous and/or discontinuous traffic styles. The detection system may allow this type of attack to pass through victims because it appears as flash-crowd traffic, which is in high demand by legitimate Internet users.

B. Mathematical Models

Based on data of arrival rates, we do need mathematical models to identify the degree of prediction. Since we categorize data into predictable and nonpredictable data, the mathematical models must be able judge the data by using a threshold. In this paper, we use *Pearson's correlation coefficient* (here after called the correlation) [16], which is defined as:

$$\rho_{X,Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X\sigma_Y} \qquad (1)$$

The correlation is used to measure dependence between two quantities (variables) X and Y with expected values μ_X and μ_Y and standard deviations σ_X and σ_Y. Both value of the standard deviations are finite and nonzero ($0 < \sigma_X < \infty$ and $0 < \sigma_Y < \infty$). One of the impressive properties of the correlation is symmetric measurement said $\rho_{X,Y} = \rho_{Y,X}$. In other words, whichever data comes first, we can still get the same result as measuring.

The correlation value is between -1 and 1 ($-1 \geq \rho_{X,Y} \geq 1$). Hence its absolute value ($|\rho_{X,Y}|$) cannot exceed 1. The absolute correlation value is 1 ($|\rho_{X,Y}| = 1$) represented by the stronger relationship between two variables called *linear dependence*. However, the absolute value ($|\rho_{X,Y}|$) from the correlation could reach zero. This does not always mean the two variables are uncorrelated. In a special case where both are normal, the uncorrelated result is also equivalent to independence. In our research, we define the data that gives us this value of 1 as predictable data with a linear form. The value of 0 ($|\rho_{X,Y}| = 0$) also defines predictable data with a symmetric form. We provide further details in the next section.

III. Problem Statement

We consider the situation where a server is overwhelmed by flash crowd flows and/or DDoS attacks as illustrated in Fig. 1. A server connects to the Internet and provides a service to public Internet users. Legitimate users do not harm the server or the service. However, the busy server could suffer a *flash* *crowd* (FC) event which is observed as a sudden high demand in service requests from Internet users. A flash crowd could overwhelm a server and create a DoS condition which results in either a delay of response or a complete crash.

DDoS attack is, however, more harmful than a flash crowd. Zombie machines (or bots) are compromised and controlled by attackers. The (botnet) attacks could be synchronized to overwhelm the victim in a specific period of time. The situation could be worse when a flash crowd merges with a DDoS attack as shown in Fig. 1. This accelerates the DoS condition to the server.

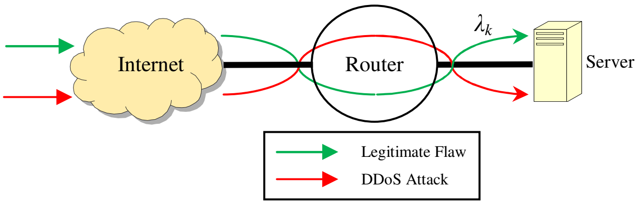


Figure 1. Accumulative arrival rate λ (packet/time interval) from *k* IP address(es).

The behavior of the bot can be detected by the victim's server by observing the predictable arrival rate. To minimize the cost of calculation, the server can observe the arrival rate (λ_k) from a high risk group of users. A study [3] found that in a botnet attack scenario, at most around 30% of bots were online at the same time for attack activities. This could possibly be an approximate number of IP addresses that need to run the bot behavior check. In particular, only 30% of user IP addresses that express high arrival rates could be checked in a period of time. This paper covers only two methods using the correlation coefficient to check arrival rates as data.

A. Method 1: Correlation between arrival rate and time

We denote X as a sample set of arrival rate ($X = \{\lambda_k\}$, where k = 0, 1, 2, … , N) and Y as a sample set of sequence number ($Y= \{k\}$ where k = 0, 1, 2, … , N). For example, $X = \{\lambda_0, \lambda_1, \lambda_2, \dots, \lambda_k\}$ and $Y = \{0, 1, 2, \dots, k\}$. Then we calculate the correlation value ($\rho_{X,Y}$) from the two variables: X and Y. The value that we expect is between -1 and 1 ($-1 \geq \rho X,Y \geq 1$).

B. Method 2: Correlation of itself arrival rate

We denote X as a sample sequence of arrival rate ($X = \{\lambda_{(2k)}\}$, where k = 0, 1, 2, … , N) and Y as a sequence number of time interval ($Y = \{\lambda_{(2k+1)}\}$ where k = 0, 1, 2, … , N). For example, $X = \{\lambda_0, \lambda_2, \lambda_4, \dots, \lambda_{(2k)}\}$ and $Y = \{\lambda_1, \lambda_3, \lambda_5, \dots, \lambda_{(2k+1)}\}$. Then we calculate the correlation value ($\rho_{X,Y}$) from the two variables: X and Y. The value that we expect is between -1 and 1 ($-1 \geq \rho_{X,Y} \geq 1$).

For both methods we calculate the correlation value and define two thresholds: upper threshold (τ_U) and lower threshold (τ_L). We can calculate these thresholds as follows:

$$\tau_U = \alpha * 1.0 \tag{2}$$

$$\tau_L = 1.0 - (\alpha * 1.0) \tag{3}$$

The value of the upper threshold must not exceed 1 and be less than the lower threshold ($1 \geq \tau_U \geq \tau_L$). On the other hand, the value of the lower threshold must not be below 0 and greater than the upper threshold ($\tau_U \geq \tau_L \geq 0$). The confidence value (α) is another adjustable value that we will discuss in the next section. As we state in our goals, these thresholds will help us to identify the dependency degree of dependency of the arrival rate data into two categories:

1) *Predictable attack rate:* The data will be classified as a predictable attack rate if the correlation value is closed to 0 or 1 as we discussed in the previous section. If the absolute correlation value is greater than the upper threshold ($\tau_U \geq |\rho_{X,Y}| \geq 1$) or less than the lower threshold ($0 \geq |\rho_{X,Y}| \geq \tau_L$), thus the data is judged as a predictable attack rate. However, we still need to define how close the correlation value can be for it to be considered a dependency arrival rate. This issue will be explained in more detail in the next section.

2) *Nonpredictable attack rate:* The data will be classified as a nonpredictable attack rate if the correlation value is not closed to 0 or 1. In other words, the data is expressed as a nonattack arrival rate and is legitimate to the service of the server. If the absolute correlation value is between the upper and lower thresholds ($\tau_U > |\rho_{X,Y}| > \tau_L$), thus the data is judged as a predictable attack rate. However, we still need to define the range of the correlation value that can be judged as an independency arrival rate. This issue will be explained in more detail in the next section.

Unfortunately, only one correlation result ($\rho_{X,Y}$) cannot determine whether arrival data is attacking or legitimate. We need a series of correlation result to confirm the situation. Hence we define $\{\rho_i\}$ is a set of the continuous results of the correlation coefficient. The i variable (i = 0, 1, 2, … , N) could be the limited number of observing the correlation value. For instance, if we want to observe the correlation for 10 values, we will have $\{\rho_0, \rho_1, \rho_2, \dots, \rho_9\}$. Each of the correlation values will be calculated with the upper threshold (τ_U) and lower threshold (τ_L) to define whether the data is predictable or not. All of these results will be calculated for an average point ($\bar{P}$) a and compared with a confidence value (α). For example, the default confidence value is 95% ($\alpha = 0.95$). The confidence value is used to judge whether this arrival data is either an attack or legitimate arrival. Finally, we decide to drop only the IP traffic that equals or is higher than the confidence value ($\bar{P} \geq \alpha$) which indicates the predictable traffic.

IV. System Optimization Analysis

In this section, we discuss optimizing variables. As we proposed in our goals, the attack detection must respond as quickly as possible after the attack reaches the victim. The computational resources also need to be minimized with simplified methods. To find the optimized variables, we analyzed the following:

A. *Size of sample analysis*

We begin from the size (k) of a sample set of arrival rates ($\{\lambda_k\}$, where k = 0, 1, 2, … , N). The question is how much sample data should be used. If the size is very small, for example k = 3, the calculation process is very quick. However, the correlation result ($\rho_{X,Y}$) may be misleading. As a result, the performance measurement gives us a high rate of false negative/positive. On the contrary, if the size is quite large, for example, k = 100, the calculation process is very slow. It also means we wait for a long time to get all the data ($\{\lambda_0, \lambda_1, \lambda_2, \dots, \lambda_{99}\}$). For example, if each k has a time slot of 0.1 second. Our defense system needs at least 10 seconds to get the first sample correlation coefficient (ρ_0). Moreover, if we observe up to 10 sample correlation data $\{\rho_0, \rho_1, \rho_2, \dots, \rho_9\}$, the detection will give us the result in at least 11 seconds (10 seconds for the first correlation data (ρ_0) plus 1 second for the remaining data $\{\rho_1, \rho_2, \rho_3, \dots, \rho_9\}$). This delay in making the decision means a weak victim could increase the possibility to crash.

B. *Correlation threshold analysis*

There are two thresholds: upper threshold (τ_U) and lower threshold (τ_L) that we consider to minimize the false positive/negative rate. To catch the predictable attacks, we need to adjust the absolute correlation value greater than the upper threshold ($\tau_U \geq |\rho_{X,Y}| \geq 1$) or less than the lower threshold ($0 \geq |\rho_{X,Y}| \geq \tau_L$). The question is what the best values are for τ_U and τ_L. If we adjust τ_U too high and τ_L too low, our detection system may fail, andas a result, the defense system may allow most attack packets to get through. On the contrary, if we adjust τ_U too low and τ_L too high, we may confront the DoS condition earlier because most packets would be considered a predictable attack. Since the two thresholds are important, the adjusted values may rely on how much confidence we have. Hence, the confidence value (α) would be calculated with these thresholds. By default, we assign the confidence value of 85% ($\alpha = 0.85$). Thus, the τ_U is 0.85 of correlation value and τ_L is 0.15 of the correlation value.

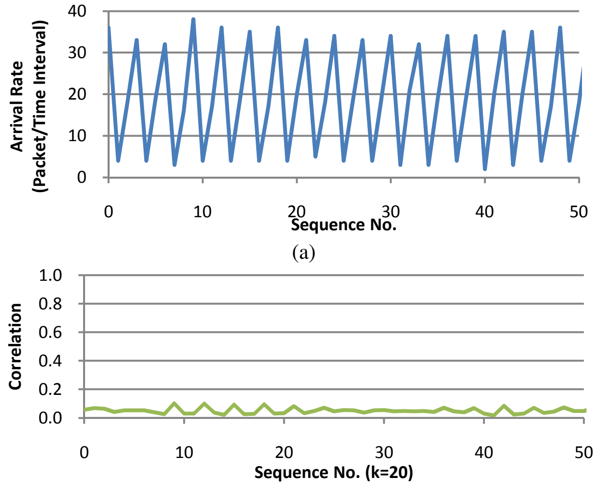
C. *Average point and confidence value analysis*

Before the final decision on the detection system, there are another two variables that need to be adjusted. One is the average point ($\bar{P}$) and another is the confidence value (α). The $\bar{P}$ is calculated from the total point (P) divided by the total number (k) of observing correlation ($\{\rho_i\}$). We borrow k from the size of the sample set of arrival rates ($\{\lambda_k\}$, where k = 0, 1, 2, … , N-1) because we need to observe the behavior as double time. For example, if we set k = 10 then we observe 10 arrival data ($\{\lambda_0, \lambda_1, \lambda_2, \dots, \lambda_9\}$). This data is calculated to only one correlation coefficient (ρ_0) which is not enough to judge the situation. Hence, we extend to observe more data as k time which create $\{\rho_1, \rho_2, \rho_3, \dots, \rho_9\}$. The optimized k has been discussed in Section 4(A).

In the next step, point (P) would be assigned based on the result of each correlation (ρ_i). If the correlation value is not lower than the upper threshold (τ_U) and is not higher than the lower threshold (τ_L), we increase P. This is because the correlation may be closed to 0 or 1 ($\rho_{X,Y} \to 0$ OR $\rho_{X,Y} \to 1$), so we need to transform the result of correlation into a marking score fashion (point P). The final step is to judge the arrival data $\{\lambda_k\}$ as either a predictable attack or not from the average point ($\bar{P}$). This process is concerned with the confidence value (α). The higher α, the more confidence we give our judgment. However, too high or too low α value leads to a high rate of negative/positive false. By default, we assign the confidence value of 85% ($\alpha = 0.85$).

V. Experimental Results

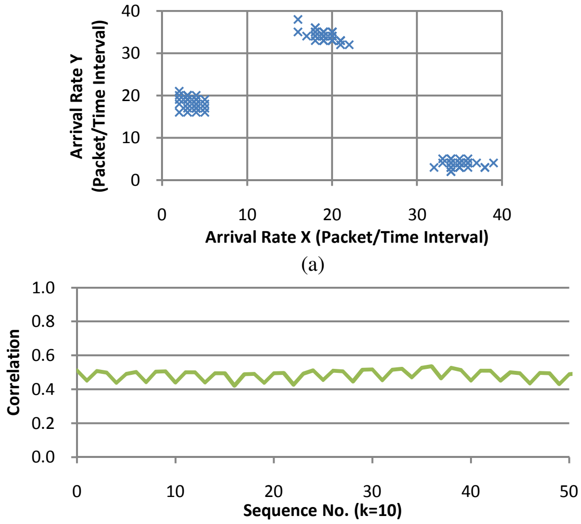
We test our methods with the generated datasets and analyze how to optimize all variables in our discrimination detection system. However, we cannot use all details of the test due to the number of results. Based on the real datasets we have, we will test them with these optimized variables using both methods. Following are examples provided with a description:



(a)

(b)

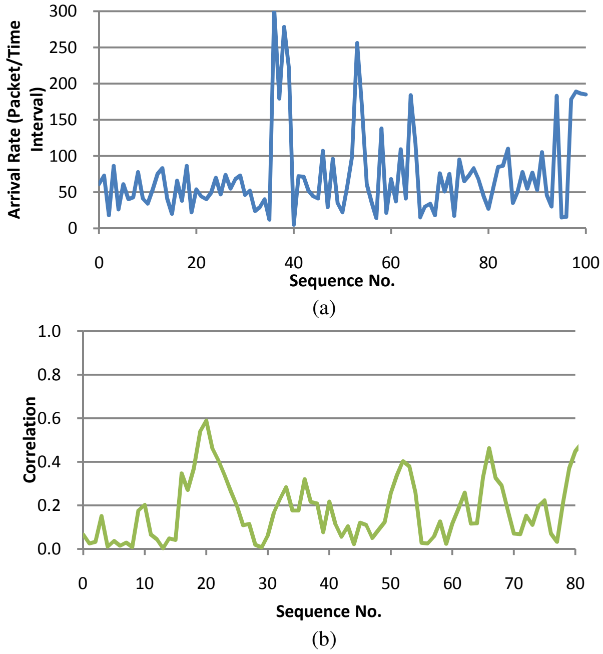
Figure 2. Experiment on sample dataset 1 (CID55) with method 1, (a) packet arrival plot, and (b) correlation from different k=20.



(a)

(b)

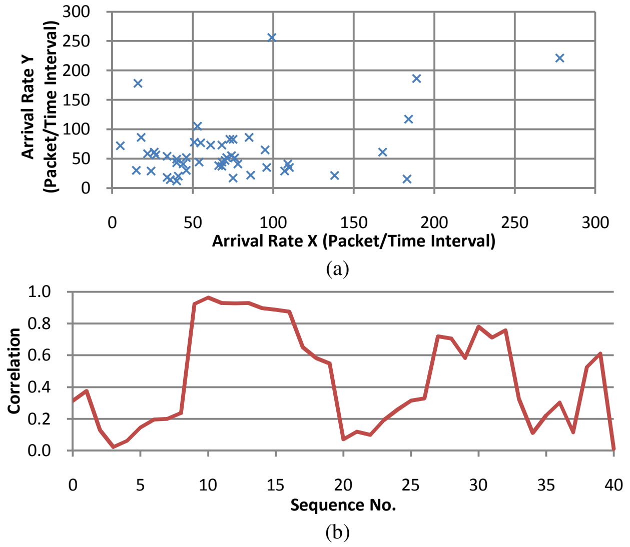
Figure 3. Experiment on generated dataset 1 (CID55) with method 2, (a) packet arrival plot, and (b) correlation with k=10.



(a)

(b)

Figure 4. Experiment on sample dataset 2 (M17060) with method 1, (a) packet arrival plot, and (b) correlation from different k=20.



(a)

(b)

Figure 5. Experiment on sample dataset 2 (M17060) with method 2, (a) packet arrival plot, and (b) correlation with k=10.

A. *Sample Dataset 1 (CID55)*

CID55 is an example of clients from a website of the World Cup 98 [17]. Its arrival rates are similar to screw attacks as depicted in Fig. 2(a). We believe this may not be a screw attack but an automatic program. Perhaps this is an effect of using Java script to update data from the website.

Now let us consider the result of correlation from both methods. The result of accumulative correlation gives us very different meanings. Method 1 detects the traffic flaw as an attack with a confidence value of 94% (as shown in Fig. 2(b)) but the other method does not (as shown in Fig. 3(b)). In this case, we must rely on the method that can detect the dependency relationship. Hence, we will not consider the correlation with different k from method 2.

As we have stated, method 1 is more reliable in this scenario. The accumulative correlation is more stable and less than 0.05 after k > 20. Hence, we consider k = 20. As a result, the maximum average point $(MAX(\bar{P}))$ is 80%, if $\alpha = 95\%$. This means that the results of correlation $\{\rho_i\}$ are $\rho_i \geq 0.05$ and $\rho_i \leq 0.95$. Because we set the confidence value too high, this scenario did not detect it as a screw attack.

The system admin may ignore this kind of low rate attack which may not be harmful to the service system. Perhaps this is a download program or Java scripts that regularly download something from the website. However, if we consider the automatic program, we could reset $\alpha = 85\%$ and then the system can detect an attack after 20 time intervals (equal to 2.0 seconds) in this scenario.

B. *Sample Dataset 2(M17060)*

M17060 is sample traffic of the client from the project of MStream attack [1]. Its arrival rates are transmitted in the random mode and are hard to detect as depicted in Fig. 4(a). As this is a high arrival rate, we expect our method to detect it as soon as possible before it can harm the server.

Now let us consider the result of correlation from both methods as shown in Fig. 4(b) and 5(b) respectively. The result of accumulative correlation tells us a similar meaning; method 1 and 2 detect the traffic flaw as an attack with 85% of the confidence threshold value. As a result, the maximum average point $(MAX(\bar{P}))$ is 100%, if $\alpha = 85\%$. This means that the results of correlation $\{\rho_i\}$ are $\rho_i \leq 0.15$ or $\rho_i \geq 0.85$.

For detection time, the two methods can detect the attack within 1.7 seconds and 2.4 seconds respectively.

C. *Sample Dataset 3(CID1387)*

CID1387 is an example of the clients from the World Cup 98 website [17]. Its arrival rates are transmitted like a flash crowd as depicted in Fig. 6(a). As this is a high arrival rate, we expect our method to detect this flow as legitimate flash crowd traffic with a low degree of harm for the server.

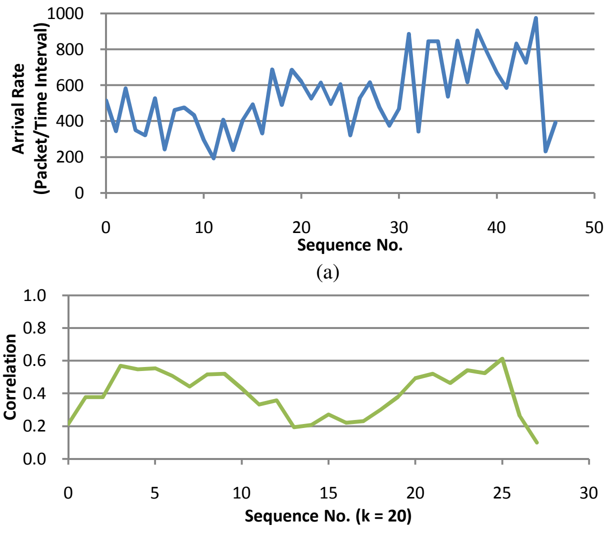
Now let us consider the result of correlation from both methods. The result of accumulative correlation as shown in Fig. 6(b) and 7(b) tells us a similar meaning; method 1 and 2 could not detect the strong relationship for the traffic flaw with 85% of the confidence threshold value. As a result, the maximum average point $(MAX(\bar{P}))$ is only 10% and 60% from both methods respectively, if $\alpha = 85\%$. This means that the results of correlation $\{\rho_i\}$ are $\rho_i > 0.15\ and\ \rho_i < 0.85$. This is because the traffic is clean and therefore, all attack request data can pass through the server.

VI. SUMMARY AND FUTHER WORK

As we stated that DDoS attack sources have a form of pattern behavior of packet transmission, with the predictability of known patterns being a very effective approach in detecting them. We propose two methods using the correlation coefficient to detect the known patterns.

Moreover, we tested these methods with generated data and a real dataset from the website of the World Cup 98 and project MStream attacks. We found the hidden predictable behavior from both datasets. The best results we achieved were 1.7 seconds and 2.4 seconds from the first method and second method respectively. We can also differentiate flash crowd traffic from DDoS attack traffic. The detection performance so far is good enough to protect the server from crashing during a DDoS attack incident. We believe that our experiment is a big step to providing the universal DDoS detection which could be implemented in any network equipment and in any Internet layer.

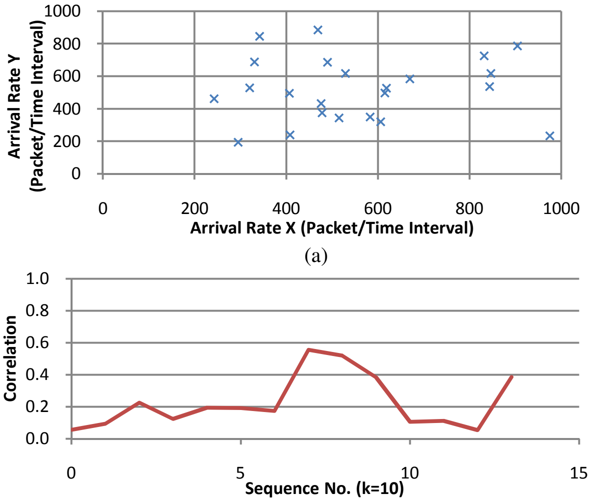
As the further works, we will test the two methods with difference packet information such as packet delay and changing rate of port number. We could test them with the real scenarios in real time. This could help us to confirm the performance from the predictability test. Moreover, a number of observing data may cause us delay in detection. We therefore need to improve our proposed methods to detect faster reducing complexity and delay.



(a)

(b)

Figure 6. Experiment on sample dataset 3 (CID1387) with method 1, (a) packet arrival plot, and (b) correlation from different *k*=20.



(a)

(b)

Figure 7. Experiment on sample dataset 3 (CID1387) with method 2, (a) packet arrival plot, and (b) correlation with *k*=10.

References

[1] MIT Lincoln Laboratory, “Lincoln Laboratory Scenario (DDoS) 1.0,” *Massachusetts Institute of Technology (MIT)*, 1999. Available: http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/2000/LLS_DDOS_1.0.html.

[2] US CERT 04, “W32/MyDoom.B Virus,” United States Computer Emergency Readiness Team, Available: http://www.us-cert.gov/cas/techalerts/TA04-028A.html, 2 Febuary 2004.

[3] M.A. Rajab, J. Zarfoss, F. Monrose and A. Terzis, “A Multifaceted Approach to Understanding the Botnet Phenomenon,” in *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, pp. 41-52, October 2006.

[4] Y. Xie and S.Z. Yu, “A Large-Scale Hidden Semi-Markov Model for Anomaly Detection on User Browsing Behaviors Networking,” *IEEE/ACM Transactions on Networking*, vol. 17, no. 1, pp. 54 - 65, February 2009.

[5] G. Oikonomou and J. Mirkovic, “Modeling Human Behavior for Defense against Flash-Crowd Attacks,” in *Proceedings of IEEE International Conference on Communications 2009 (ICC '09)*, pp. 1 - 6, 11 August 2009.

[6] Y. Xie and S.Z. Yu, “Monitoring the Application-Layer DDoS Attacks for Popular Websites,” *IEEE/ACM Transactions on Networking*, vol. 17, no. 1, pp. 15-25, February 2009.

[7] Y. Chen and K. Hwang, “Spectral Analysis of TCP Flows for Defense against Reduction-of-Quality Attacks,” in *Proceedings of the 2007 IEEE International Conference on Communications (ICC’07)*, pp. 1203–1210, June 2007.

[8] F. Yi, S. Yu, W. Zhou, J. Hai and A. Bonti, “Source-Based Filtering Algorithms against DDoS Attacks,” *International Journal of Database Theory and Applications,* vol. 1, no. 1, pp. 9-22, 2008.

[9] L. Khan, M. Awad and B. Thuraisingham, “A new intrusion detection system using support vector machines and hierarchical clustering,” *The International Journal on Very Large Data Bases (The VLDB Journal)*, vol. 16, no. 4, pp. 507-521, Springer-Verlag, New York, October 2007.

[10] L. Feinstein, D. Schnackenberg R. Balupari and D. Kindred, “Statistical Approaches to DDoS Attack Detection and Response,” in *Proceedings of the DARPA Information Survivability Conference and Exposition,* vol. 1, IEEE CS Press, 22-24 April 2003, pp. 303–314.

[11] S. Yu, T. Thapngam, J. Liu, S. Wei and W. Zhou, “Discriminating DDoS Flows from Flash Crowds Using Information Distance,” in *Proceedings of the 3rd IEEE International Conference on Network and System Security (NSS’09)*, 18-21 October 2009.

[12] Y. Carlinet, O. Cherkaoui, F. Dressler, C. Ehinger, A. Fadlallah, G. Muenz, M. Mußner, O. Paul, A. Serhrouchni, M. Sloman, and S. Yusuf, “Distributed Adaptive Security by Programmable Firewall,” *DIADEM Firewall Consortium*, retrieved 6 September 2008, Available: http://www.diadem-firewall.org/documents/Diadem%20Firewall%20-%20D3%20-%20Attack%20Requirements%20Specification.pdf, June 2004.

[13] Y. Chen and K. Hwang, "Collaborative detection and filtering of shrew DDoS attacks using spectral analysis,” *Journal of Parallel and Distributed Computing*, vol. 66, no. 9, pp. 1137-1151, September 2006.

[14] T. Tuncer and Y. Tatar, “Detection SYN Flooding Attacks Using Fuzzy Logic,” in *Proceedings of International Conference on Information Security and Assurance (ISA’08)*, pp. 321-325, 24-26 April 2008.

[15] A. Kuzmanovic and E. Knightly, “Low-Rate TCP –Targeted Denial of Service Attacks (The Shrew vs. the Mice and Elephants),” in *Proceedings of ACM SIGCOMM 2003*, Kalrushe, Germany, pp. 75-86, August 2003.

[16] E. Kreyszig, *Advanced Engineering Mathematics*, 9th ed., Wiley, Singapore, 2006.

[17] M. Arlitt and T. Jin, “1998 World Cup Web Site Access Logs,” August 1998. Available: http://www.acm.org/sigcomm/ITA/.