Stepping Stone Detection under Timing Perturbations through the Uniform Distributed Random Delay

Koohong Kang
Dept. of information and communications Engineering,
Seowon University
Cheongju, Republic of Korea
e-mail: khkang@seowon.ac.kr

Jungtae Kim and Ikkyun Kim
Information Security Research Division,
Electronics and Telecommunications Research Institute
Daejeon, Republic of Korea
e-mail: {jungtae_kim,ikkim21}@etri.re.kr

***Abstract*—Even if many research works for detecting the interactive stepping stones have been presented, it is a still very challenging problem due to the intruder evasions, such as timing perturbations and adding meaningless packets called the chaff. Instead of using more elaborate techniques to timely perturb the streams crossing over stepping stones, many previous works have been exploiting the uniformly distributed random delays. In this paper, we revisit the de-synchronization problem between the original and transformed streams at a stepping stone when we use a simple uniform distribution for timing perturbations. To do so, we present a limitation of the range of uniform distribution for adding the local timing jitters in terms of the user's maximum tolerable delay. In particular, we simulate the delay distribution of the perturbed stream in terms of $\mathrm{Pareto}(\alpha, \beta)$, which represents the packet inter-arrivals. We also define a simple metric to determine the correlation between two traffic streams for detecting the stepping stones. Finally, we show that our detection algorithm is robust to the timing perturbations within the maximum tolerable delay.**

Keywords-stepping stones; timing perturbations; evasion; interactive services.

I. INTRODUCTION

Intruders on the Internet often attack their targets indirectly by staging their attacks through intermediate hosts known as stepping stones to make it complicated to trace them. For example, an attack may traverse a sequence of hosts through a chain of interactive connections using Telnet, Rlogin, or secure shell (SSH). Over the past decades, several approaches have been introduced to find the interactive stepping stones. Zhang and Paxson [1] proposed the first timing-based method that uses the packets' arrival time information, and Yoda and Etoh [2] defined the minimum average delay gap between the packet streams of two connections as the deviation. Since then, many research works [3][5] have been presented using only the packet timing characteristics because these systems can be used to find steeping stones even when the traffic is encrypted. These algorithms are based on the timing information, however, are all vulnerable to the active timing perturbation by the attacker; that is, the intruder can possibly evade the detection systems by modifying the packet timing information at the stepping stones [4]-[10].

Donoho et al. [4] first discussed evasions that consist of the local jittering of packet arrival times. They also assume that an intruder has the maximum tolerable delay that an attacker is willing to introduce since humans are not able to work effectively over the interactive connections with a very long latency. Under the bounded delay assumption, Blum et al. [9] and He/Tong [10] extended the work of Donoho et al. [4] to correlate between two streams. They based on the packet counting process of the bidirectional streams (incoming and outgoing streams at a monitoring point) with a packet-conservation constraint; that is no packets are generated or dropped at the stepping stones. Yang/Huang [11] and Yang/Zhang [12] monitored the Send and Echo packets at the incoming and outgoing session of a host, and then compute the number of RTTs for both sessions. If the difference between the two numbers of RTTs is bounded, then it indicates that the host is used as a stepping-stone. However, the pair-wise (incoming and outgoing) monitoring at a single point could make the proposed system unrealistic in the national/world-wide Internet due to traffic asymmetric induced by routing policies [13]; that is, the packet streams between two endpoints follows the different physical links between intermediate nodes for both forward and reverse direction. In this paper, our main contribution is to propose a passive network-based approach to correlate between two streams without considering the directions of streams.

To meet the maximum tolerable delay of a transformed stream from the original inbound stream, Donoho et al. [4] used the dyadic block reshuffling. However, many researchers [5][6][7] are still considering a simple uniform random delay to perturb the timing information because the attackers can embed a simple delay routine into the pseudo-tty programs for interactive services. In this paper, we revisit the de-synchronization problem between the original and transformed streams at a stepping stone when we use a simple uniform distribution for timing perturbations. That is, we present a limitation of the uniform distribution ranges for adding a local timing jittering in terms of the user's maximum tolerable delay. We also propose a practical approach to correlate between connections for finding the stepping stones. We will show that our detection algorithm is robust to the timing perturbations within the maximum tolerable delay because the total time interval of the ON times (a burst of packets) or the OFF times (no packets) is less fluctuated compared with the packet level jittering.

The rest of this paper is organized as follows. Section 2 discusses the related works in timing perturbations. In Section 3, we model the uniform distributed time delay, and

 ISBN: 978-1-61208-516-6

then simulate the delay distribution of the perturbed stream in terms of Pareto(α, β), which represents the packet inter-arrivals. In Section 4, we define a simple metric to determine the correlation between two traffic streams for detecting stepping stones, and then provide the experimental results. Finally, we conclude in Section 5.

II. RELATED WORKS

Donoho et al. [4] indicated that there are theoretical limits on the ability of attackers to disguise their traffic using evasions for a sufficiently long connection. They prepared a transformed stream using the dyadic block reshuffling which supports their assumption of a maximum delay tolerance; that is, they keep the same number of packets for each fixed time bin in the original and transformed streams, and times for packets in transformed stream are chosen uniformly at random within the time bin.

Venkateshaiah and Wright [8] proposed a simple buffering technique that could be used by an attacker on a stepping stone to evade detection, in which the transformed stream generates a constant rate traffic similar to the characteristics of a multimedia stream such that the timing correlations between the incoming original stream and the outgoing transformed stream do not exist anymore. Hence, they used a watermark-based timing analysis algorithm for detecting stepping stones. Even if the intruders install a crafty program for the interactive services, which introduces delays to make the incoming and outgoing streams to have a different timing characteristic as similar to the above two studies[4,8], we can easily expect a fact that the intruders embed a simple random delay routine into the existing interactive service programs. Wang and Reeves [5] used that the random delays added by the attacker are up to a maximum 1400ms timing perturbation. Zhang et al. [7] also added uniform distributed delays to each original flow for their experiments. In particular, they consider 10 different kinds of uniform delays, and their maximum delays increase from 2 to 20 seconds by incrementing 2 seconds gradually. However, we will identify the fact that these delays over 500 milliseconds timing perturbation are unrealistic in the real world. Peng et al. [6] experimented with 9 different timing perturbation variables, which are uniformly distributed with a maximum delay from 0 to 8 seconds. As mentioned earlier, Donoho et al. [4] noted that the incoming and outgoing streams become unboundedly out-of-sync if we simply add random delays to make a series of the time perturbed streams. In this paper, we also systematically review the de-synchronization problem, and simulate how much the delays can be added when we use a uniformly distributed random delay within the boundary of the maximum tolerable delay.

III. TIMING PERTURBATIONS

A. Delay Modeling

Most network operating systems provide an interactive service client and server, such as Telnet, Rlogin and SSH. These clients and servers are small executable programs that allow a local computer to access services and programs on a remote computer. An attacker who has a complete control over the compromised stepping stone should install a rogue application that receive and forward the incoming streams by adding some delays to make the timing perturbations. Fig. 1 shows a simple block diagram showing an example of these rogue programs working as both a client and server functions, where we assume that the attacker add an uniform distributed delay $(0, R)$.

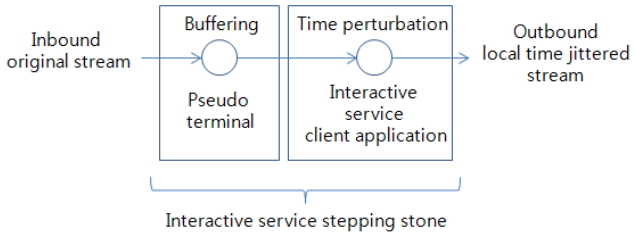

Figure 1. Processing of interactive services at a stepping stone.

Assuming the buffering and processing overheads are negligible compared with the intentional delay time according to the intruder's perturbation, we can depict the packet arrivals and departures at a stepping stone as shown in Fig. 2. Each original packet's arrival time from the original stream received is $t_1, t_2, \cdots, t_n$ at the stepping stone and its corresponding packet's departure time to the transformed stream that outgoing out from the stepping stone is $u_1, u_2, \cdots, u_n$ respectively.

We note that the time instants of packet departures depend on the amount of uniform distributed delay $(0, R)$ as well as the packet inter-arrival time distribution of the incoming stream. For example, the departures of packets 1, 2, and 3 shown in Fig. 2 are determined by the arrival times of each packets and the randomly chosen delay times ($(u_1 - t_1) \sim \text{UNI}(0, R)$) (we call this *Case-I*). However, the departure of packet 4 is determined only by the randomly chosen delay time because the corresponding incoming packet 4 already arrived before time u_3 so that the packet stored at the buffer is waiting for departure ($(u_4 - u_3) \sim \text{UNI}(0, R)$) (we call this *Case-II*).

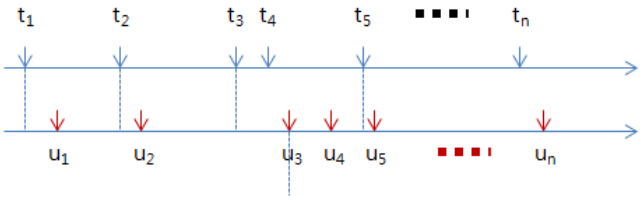

Figure 2. An example of the packet arrivals and departures at a stepping stone.

The notion of maximum tolerable delay was first discussed by Donoho et al. [4]; that is, the attacker trying to evade a stepping stone detection would not be able to work effectively over the interactive connections with a very long latency. Hence, the time difference δ_i of the arriving and departing packet i at a stepping stone must be within a time interval Δ:

$$\delta_i = u_i - t_i \leq \Delta$$

As we can expect, δ_i depends on the distribution of packet inter-arrival time as well as the distribution of random delay. Paxson and Floyd [14] showed that the users' typing patterns of Telnet service fit very well to a Pareto distribution with a shape parameter 0.9 or 0.95. In this paper, we also use the Pareto distribution instead of the exponential distribution because the exponential distribution results in seriously underestimating the longer inter-arrivals (burstiness) of interactive services due to the heavy tailed property of their inter-arrival times.

B. Simulation Results

We evaluate the time differences δ_i by simulation while generating one million packets using the Pareto distribution with $\alpha = 0.1$ and $\beta = 0.9$, and try to figure out the fluctuation levels of the δ_i with a diverse uniform distribution parameter R. We also use the different seeds of random number generations for each simulation throughout this paper. Assuming the maximum tolerable delay as 10 seconds in this paper, we can determine the maximum R of the uniform distribution from the simulation results.

As shown in Fig. 3, the δ_i can be limited to 1 sec at $R = 0.2$ sec, 3 sec at $R = 0.4$ sec, 5 sec at $R = 0.5$ sec, and 45 sec at $R = 1$ sec respectively. We figured out that the δ_i increase abruptly from $R = 0.5$ sec because the Case-II events happen more frequently than the Case-I events, which means that the delays of the previous packets are accumulated into the delay of the current packet.

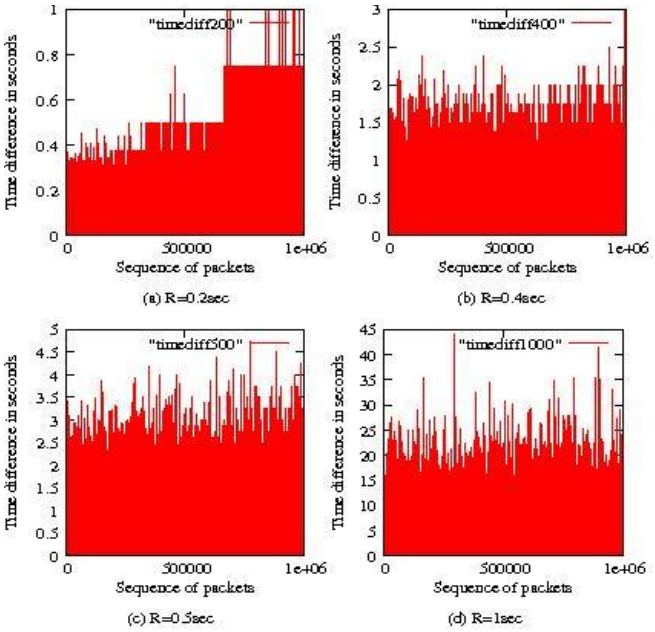

Figure 3. Time differences of each packet on incomming stream and its corresponding packet on outgoing stream with diverse R= 0.2, 0.4, 0.5, and 1 seconds.

Hence, for the timing perturbations, we should avoid of using a delay more than 0.5 sec for parameter R with the uniform distribution, because a longer delay will obviously violate the maximum tolerable delay constraint.

We also simulate the 'similarity' of the ON (a burst of packets) and OFF (idle) sequences between the incoming stream and the time perturbed outgoing stream. For this purpose, we generate flows with a group of the consecutive packets whose inter-arrival times are less than a threshold called the inactive time out. In other words, in case when a packet is generated after the inactive time out, then a new flow is generated. Hence, the duration of flow becomes an ON time, and the inter-flow time becomes an OFF time. According to the characteristics of packet inter-arrival times, a single stream possibly consists of a group of flows. Finally, we can obtain the sequences of ON and OFF times of the incoming stream and the outgoing stream.

In order to determine the 'similarity' between two streams, we consider their random walks. That is, every time the stream is on an ON (or OFF) time, we walk positively (or negatively) as much as the corresponding ON (or OFF) duration to the y-axis. Fig. 4 shows the random walks of different $R = 0.2, 0.4, 1.0,$ and 2.0 sec. From Fig. 4 (a) and (b), two lines of the random walks of the original and transformed streams nearly overlap. However, the differences over $R = 1.0$ second (see Fig. 4 (c) and (d)) are getting greater as R increases. Consequently, we define a metric for the similarity between two streams as follows,

$$\mathrm{MA_{ON}} = \frac{\text{Total duration of ON times mactched between two streams}}{\text{Total duration of ON times about the reference stream}}$$

We can also define MA_{OFF} similar to the MA_{ON}. Table I shows a simulation result for the diverse R of $\mathrm{UNI}(0, R)$.

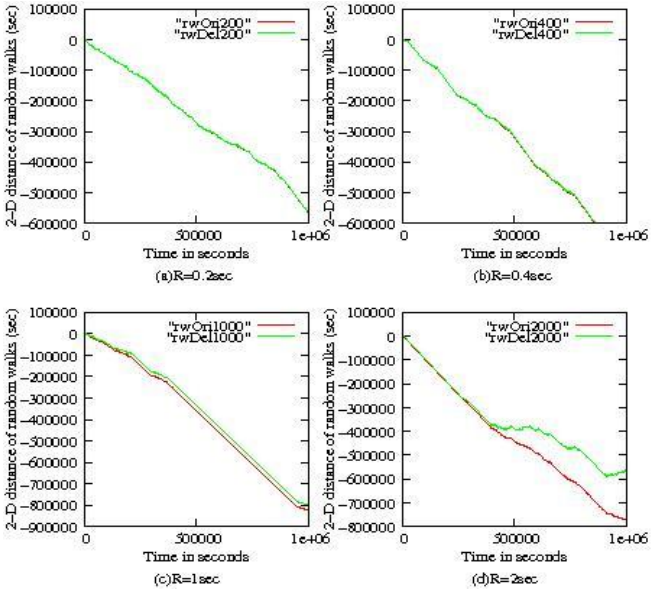

Figure 4. 2-dimensional random walks of the ON and OFF sequences of the original and transformed streams with diverse R= 0.2, 0.4, 1.0, and 2.0 seconds (rwOri – original stream, rwDel – transformed stream).

We use different seeds of the random number generations for each simulation; hence there are different lengths of the ON and OFF time sequences called as the fingerprints (FPs).

As shown in the Table I, in particular the lengths of fingerprints of the transformed streams are getting smaller as R increases; that is because the delays are accumulated, and then the packets in the transformed stream tend to get together side-by-side within a widened burst interval. However, the metrics MA_{ON} and MA_{OFF} for measuring the correlation between two connections are still very high if R is less than 1 second.

TABLE I. SIMULATION RESULTS

	UNI(0, *R*)			
	R=0.2sec	***R=0.4sec***	***R=1.0sec***	***R=2.0sec***
No. of original FP	31,501	31,259	31,833	31,741
No. of delayed FP	31,277	30,079	25,737	7,133
Total ON time	527,532.61	526,398.55	525,385.41	526,733.59
Total matched ON time	525,434.43	524,112.00	522,339.52	526,007.11
MA_{ON}	0.998	0.995	0.994	0.998
Total OFF time	2,128,542.77	9,797,486.78	17,751,016.03	4,820,881.39
Total mathed OFF time	2,125,116.78	9,785,916.63	17,680,996.00	4,224,253.90
MA_{OFF}	0.998	0.998	0.996	0.899

For example, MA_ON=0.994 and MA_OFF=0.996 when R=1 second which is beyond our considering limitation for the time delays due to the fact that R over 1 sec triggers an accumulated packer delay up to 45 seconds (Fig. 3-d).

IV. DETECTION ALGORITHM AND ITS PERFORMANCE EVALUATIONS

In this paper, we evaluated two simple metrics of the MA_{ON} and MA_{OFF} for measuring the correlation between two connections. We declare that any two connections are on the same connection chain if the MA_{ON} and MA_{OFF} between them are greater than a given threshold.

Detect-Attacks (θ, FP_{ref})
Obtain a set S in which every connection has its connection time interval overlapping the connection time interval of the reference connection;
Prepare the FP for every connections in the set S;
For every connections in the set S
Compute MA_{ON} and MA_{OFF};
If $MA_{ON} > \theta$ and $MA_{OFF} > \theta$
Alert Attack;

Figure 5. Algorithm for stepping stone detection.

To figure out the performance of our proposed algorithm shown in Fig. 5, we use the data set of the Auckland-IV traces in NLANR PMA Daily Traces Archive [15]. We organized the data set into four parts such as the traffic coming into the University and the traffic originating from the University, and two monitoring days separated by a monitoring interruption. We extract 8,648 unidirectional connections such as Telnet, SSH, and Rlogin connections which must have their own pairs (up-stream/ down-stream, or client-to-server/server-to-client) on their other directions, and last for more than two minutes of their connection intervals, and have more than 10 ON and OFF times values as their fingerprints.

To figure out the false rates, we chose one connection from 8,648 connections, and then calculate MA_{ON} and MA_{OFF} between the chosen connection and the others. If we cannot find the selected corresponding connection on the opposite direction using our proposed approach, we count the false negative; for example, if the given connection is a client-to-server connection of an interactive service incoming into the University, then we have to find its corresponding server-to-client connection originating from the University. Moreover, if we find stepping stone connections that should not be correlated with the given connection, we count the false positive; that is, we count the stepping stone connections which have start and stop times such that they do not overlap with the connection interval of the given connection.

Table II shows the experimental results of the false negative and positive rates with the varying threshold θ from 0.5 to 0.95 respectively. In case of finding the false positive (or negative), there are totally 74,779,256 (or 8,648) comparisons for calculating correlations between two connections. From Table II, we can set the $\theta = 0.75$ in order to keep the false negative and false positive rates under 1%. We evaluated that the value of $\theta = 0.75$ is enough to detect the corresponding time perturbed stream for a given original stream as explained in Table I.

TABLE II. FALSE POSITIVE AND FALSE NEGATIVE OF OUR PROPOSED ALGORITHM WITH DATA SET OF AUCKLAND-IV TRACES

Threshold θ	***False Negative (%)***	***False Positive (%)***
0.5	0.1388	7.3047
0.55	0.1504	4.9629
0.6	0.2314	3.2712
0.65	0.3239	2.1379
0.7	0.4628	1.4183
0.75	0.8677	0.9439
0.8	1.3999	0.6382
0.85	2.4297	0.4192
0.9	4.7437	0.2521
0.95	9.8923	0.0782

We also evaluate the performance of our algorithm under the time perturbations. For this purpose, we add 4 different timing perturbations to the selected connections from the Auckland-IV traces, which are uniform distributed with a maximum delay 0.2, 0.4, 0.5, and 1 second. These perturbations could make the correlated streams uncorrelated, and the uncorrelated streams correlated. We chose each connection from these 8,648 original connections, and then calculate the correlations between the selected connection and each timely perturbed connection. We should find 17,284 stepping stone connections and 74,770,608 non-stepping stone connections. Figure 6 and 7 show the false

 ISBN: 978-1-61208-516-6

negative rates and false positive rates of our proposed algorithm, respectively.

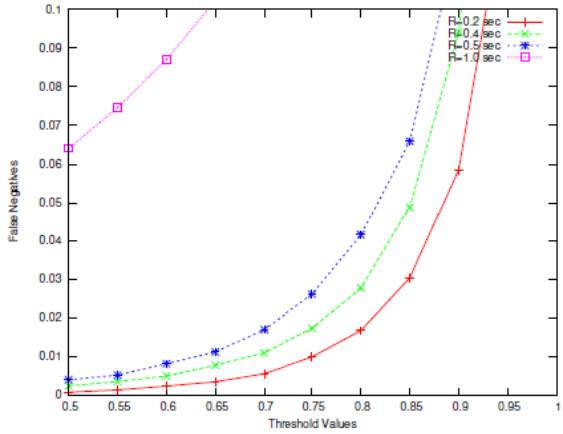

Figure 6. False negative rate with different timing perturbations.

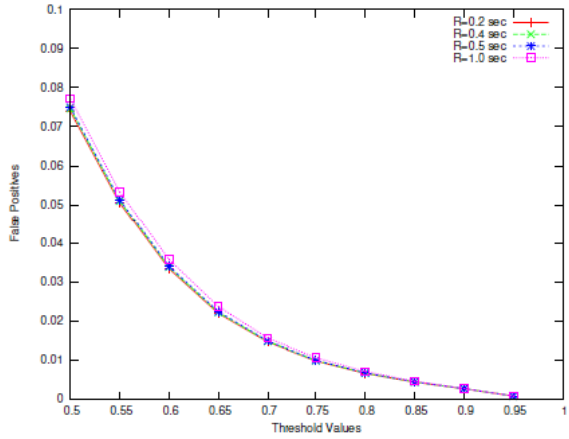

Figure 7. False positive rate with different timing perturbations.

From Figs. 6 and 7, our detection algorithm can achieve about 1-2% false rates at $\theta = 0.75$ when the maximum R of uniform random delay is less than 0.4 second.

V. CONCLUSION

In this paper, we have considered a de-synchronization problem between the original and transformed streams at a stepping stone when we use a simple timing perturbations with the uniform distributed random delays for evasions of the stepping stone detections. From the simulation with the Pareto distribution for packet inter-arrivals, we showed the delay bounds for using the uniform distribution random delays in terms of the maximum tolerable delay. We have also presented a simple metric to detect the stepping stones under these evasions. In particular, we showed the proposed detection algorithm works efficiently under the effects of time perturbations because the packet level jittering is insignificant for calculating the total time interval of the ON or OFF times. Finally, we presented the false rates of our detection algorithm through the experiment with a real data.

ACKNOWLEDGMENT

This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (No.B0101-15-1293, Cyber-targeted attack recognition and traceback technology based on the long-term historic analysis of multi-source data.

REFERENCES

[1] Y. Zhang and V. Paxson, "Detecting Stepping Stones," Proc. of the 9th USENIX Security Symposium, pp. 171-184, August 2000

[2] K. Yoda and H. Etoh, "Finding a Connection Chain for Tracing Intruder," In Computer Security-ESORICS 2000, pp. 191-205, 2000

[3] X. Wang, D. S. Reeves, and S. F. Wu, "Inter-Packet Delay Based Correlation for Tracing Encrypted Connections Through Stepping Stones," In Computer Security-ESORICS 2002, pp. 244-263, 2002

[4] D. L. Donoho et al., "Multiscale Stepping-Stone Detection: Detecting Pairs of Jittered Interactive Streams by Exploiting Maximum Tolerable Delay," In Recent Advances in Intrusion Detection, Springer Berlin Heidelberg, pp. 17-35, 2002

[5] X. Wang and D. S. Reeves, "Robust Correlation of Encryped Attack Traffic Through Stepping Stones by Manipulation of Interpacket Delays," Proc. of the 10th ACM conference on Computer an dcommunications security, pp. 20-29, Oct. 2003,

[6] P. Peng, P. Ning, D. S. Reeves, and X. Wang, "Active Timing-Based Correlation of Perturbed Traffic Flows with Chaff Packets," In Distributed Computing Systems Workshops, pp. 107-113, 2005

[7] L. Zhang, A. G. Persaud, A. Johnson, and Y. Guan, "Detection of Stepping Stone Attack under Delay and Chaff Perturbations," In Performance, Computing, and Communications Conference, pp. 247-256, April 2006

[8] M. Venkateshaiah and M. Wright, "Evading Stepping Stone Detection under the Cloak of Streaming Media," CAE@UTA Technical Report, 2007

[9] A. Blum, D. Song, and S. Venkataraman, "Detection of interactive stepping stones: Algorithms and confidence bounds," In Proc. Conf. Adv. Intrusion Detection (RAID), pp. 258-277, 2004

[10] T. He and L. Tong, "Detecting Encrypted Stepping-Stone Connections," IEEE Transactions on Signal Processong, vol. 55, no. 5, pp. 1612-1623, May 2007

[11] J. Yang and S. Huang, "Mining TCP/IP Packets to Detect Stepping-Stone Intrusion," Journal of Comupters and Security, Elsevier Ltd., vol. 26, pp. 479-484, 2007

[12] J. Yang and Y. Zhang, "RTT-based Random Walk Approach to Detect Stepping-Stone Intrusion," In Proc. of International Conference on Advanced Information Networking and Applications, pp. 558-563, 2015

[13] W. John, M. Dusi, and K. C. Claffy, "Estimating Routing Symmetry on Single Links by Passive Flow Measurements," In Proc. of the 6th International Wireless Communications and Mobile Computing Conference, pp. 473-478, 2010

[14] V. Paxson and S. Floyd, "Wide-Area Traffic: The Failure of Poisson Modeling," IEEE/ACM Transactions on Networking, vol. 3, no. 3, pp. 226-244, June 1995

[15] WITS: Waikato Internet Traffic Storage, available: http://wand.net.nz/wits/auck/4/auckland_iv.php, retrieved: Feb.2016

 ISBN: 978-1-61208-516-6