Article

Not peer-reviewed version

An Effective Approach for Stepping-stone Intrusion Detection Resistant to Intruders' Chaff-Perturbation via Packet Crossover

Lixin Wang *, Jianhua Yang, Jae Kim, Peng-Jun Wan

Posted Date: 2 August 2023

doi: 10.20944/preprints202308.0200.v1

Keywords: stepping-stone intrusion; connection chain; session manipulation; chaff-perturbation; packet crossover

Article

An Effective Approach for Stepping-Stone Intrusion Detection Resistant to Intruders' Chaff-Perturbation via Packet Crossover

Lixin Wang [1,*], Jianhua Yang [1], Jae Kim [1] and Peng-Jun Wan [2]

[1] TSYS School of Computer Science, Columbus State University, GA, USA; wang_lixin@ColumbusState.edu
[2] Department of Computer Science, Illinois Institute of Technology, IL, USA; wan@cs.iit.edu
* Correspondence: wang_lixin@ColumbusState.edu; Tel.: (001)-706-507-8190

Abstract: Today's intruders usually send attacking commands to a target system through several stepping-stone hosts, in order to reduce the chance of being detected. With stepping-stone intrusion (SSI), the intruder's identity is hidden behind a long interactive chain of hosts and very hard to detect. An effective approach for SSI detection (SSID) is to estimate the length of the chain. This type of method is called network-based SSID. Most existing network-based SSID worked effectively only when intruders' session manipulation was not present. These known SSID algorithms are either weak to resist intruders' chaff-perturbation manipulation or having very limited capability in resisting attacker's session manipulation. This paper develops a novel network-based SSID algorithm resistant to intruders' chaff-perturbation by using packet crossover. Our proposed SSID algorithm is simple and easy to implement as the number of packet crossovers can be easily computed. We conduct rigorous technical proofs to verify the correctness of our proposed algorithm. The experimental results show that our proposed SSID algorithm works effectively and perfectly in resisting intruders' chaff-perturbation up to 50% chaff rate.

Keywords: stepping-stone intrusion; connection chain; session manipulation; chaff-perturbation; packet crossover

1. Introduction

With stepping-stone intrusion (SSI), an attacker builds a chain of stepping-stone hosts (see Figure 1), uses SSH or telnet to login in turn to these stepping-stones and then launches the attack. In Figure 1, Host 0 serves as the attacker host. The intruder then remotely logins in turn to the hosts Host 1, Host 2, . . ., Host i-1, Host i, Host i+1, . . ., and Host N -1 that serve as stepping-stones. The last Host N in the chain represents the victim target system. To detect SSI, any stepping-stone host between the attacker and the victim can serve as the sensor where a packet sniffer program (E.g., TCPdump etc.) is running to capture network traffic. In Figure 1, we assume that Host i is the sensor with a packet sniffer program installed. The left part of the chain from the intruder's host to the sensor host is referred to as an upstream sub-chain, whereas the right part of the connection chain from the sensor to the victim is referred to as the downstream sub-chain.

SSI detection (SSID) is to determine whether a stepping-stone host is used by a hacker for intrusion. If there is at least one relayed pair between all the incoming connections and all the outgoing connections of the sensor, then it is highly suspicious that the sensor host is used as a stepping-stone and the session is an intrusion. Stepping-stone intrusions are very hard to detect as the intruder is hidden behind a long TCP/IP connection chain. Since each interactive TCP session between a client and a server is independent of other sessions even though the sessions may be relayed, so accessing a server via multiple relayed TCP sessions can make it much harder to tell an intruder's identity.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 August 2023 doi:10.20944/preprints202308.0200.v1

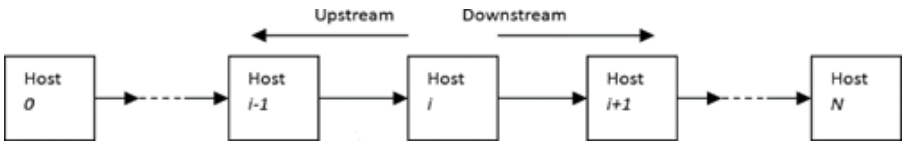

Figure 1. A sample of a connection chain.

One type of SSID approach is to compare all the outgoing connections with all the incoming connections of the sensor to see if a relayed pair of connections is present. This type of SSID approach only uses the sensor host for detection, thus is referred to as the host-based SSID [2,4,5,12–18]. It is well-known that most Web applications as well as cloud computing applications usually employ stepping-stone hosts to gain access to a remote server such as a database server. Therefore, high false-positive errors are likely unavoidable when host-based SSID approach is used for detection.

To reduce high false-positive errors for SSID, another type of SSID method was proposed by estimating a connection chain length. This type of SSID method is referred to as the network-based SSID [1,7,9,11]. That is, this type of method focuses on estimating the number of connections in the chain (referred to as the length of the chain) from the attacker host to the victim (as shown in Figure 1). If an attacker uses two or more stepping-stone hosts to launch an attack, then the number of connections from the attacker host to the victim is at least three. It is well-known that most legitimate applications such as Web applications and cloud computing applications rarely utilize two or more relayed stepping-stones to gain access to a remote server. Thus, it is most likely the session is an intrusion when a machine is accessing a remote server through two or more stepping-stone hosts.

Now let us give a thorough review of all the existing network-based SSID algorithms that perform intrusion detection via estimating the length of a connection chain. The first network-based SSID method was proposed by Yung et al. [11] in 2002. This approach computed a connection chain length by calculating the ratio of the Send-Echo RTT over the Send-Ack RTT. According to Yung's method, a Send-Echo RTT reflects the connection chain length from the sensor host to the victim host. Whereas a Send-Ack RTT represents the length of one hop connection from the sensor host to its adjacent machine on the downstream side. The problem with Yung's SSID algorithm is that it produced very high false-negative errors because the acknowledgement packets were used in the chain length estimation. The issues of Yung's SSID algorithm were discovered and described in [7] by J. Yang et al.

The work [7] was the 2nd network-based SSID algorithm and proposed in 2004. The step function method was used to estimate a connection chain length in [7]. This paper resolved the issues of Yung's SSID algorithm in [11] by setting up the connection chain in a different way. With this improvement, each Send packet can be matched with a corresponding Echo in [7]. Thus, the false-negative errors for SSID were significantly reduced in [7], compared to Yung's SSID algorithm proposed in [11]. Unfortunately, the step-function method proposed in [7] was only performing effectively in a local area network.

Under the Internet environment, Yang et al. tried to fix the issue of the SSID algorithm proposed in [7] and proposed a greedy packet matching SSID algorithm in [8]. However, SSID algorithm proposed in [8] can only match very few packets, and thus this method did perform efficiently.

To overcome these issues existing in [8], a clustering and partitioning data mining SSID approach was proposed by Yang et al. in [9]. The packets' round trip times were computed by using clustering and partitioning data mining algorithm – the maximum-minimum distance clustering algorithm. With this method, every Send packet was accurately matched through looking at all the possible Echoes for this Send. The number of connections in the chain was determined by the number of clusters produced by the maximum-minimum distance clustering algorithm. But the SSID method proposed in [9] has to capture a huge number of TCP packets, which makes the processing and analysis of the captured packets not efficient. Thus, the SSID method proposed in [9] is not efficient, taking the packets processing time into consideration.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 August 2023 doi:10.20944/preprints202308.0200.v1

To overcome the issue with the detection method proposed in [9], a k-Means clustering based SSID algorithm was proposed by Wang et al in [1]. With this method, packets processing and analysis are efficient as this method did not require capture a large number of packets. Due to the nature of the k-Means clustering algorithm, it requires that a connection chain length must be predetermined. Therefore, the usage and capability of the SSID algorithm proposed in [1] are limited. When there are a lot of outlier values in the packets RTTs, the SSID algorithm proposed in [1] won't perform effectively.

In 2022, a network-based SSID algorithm using packet crossover was proposed by Wang et al. in [20]. The downstream length of a connection chain was estimated by analyzing the packet crossover ratios in [20]. The work [20] also verified that a downstream connection chain length strictly increases as the packet crossover ratios. However, the SSID algorithm proposed using packet crossover in [20] was not resistant to intruders' chaff-perturbation.

This paper addresses the problems with the SSID method proposed in [20]. The contributions of this paper are summarized below: 1) This paper develops a novel network-based SSID algorithm that is resistant to intruders' chaff-perturbation by using packet crossover; 2) the proposed SSID algorithm is simple and easy to implement as the number of packet crossovers can be easily computed; 3) the proposed SSID algorithm is network-based and generates very low false-positive errors; 4) rigorous technical proofs are conducted to verify the correctness of our proposed SSID algorithm; the proposed SSID algorithm works effectively and perfectly in resisting intruders' chaff-perturbation up to 50% chaff rate based on our experimental results; to the best of our knowledge, this is the first network-based SSID algorithm that can effectively estimate the length of a connection chain as well as resist intruders' chaff-perturbation up to 50% chaff rate.

The remaining of this paper is organized as follows. In Section 2, we introduce some basic concepts and preliminary knowledge which can help readers understand our proposed SSID algorithm in this paper. In Section 3, we give a proposition that asserts the relationship between the downstream sub-chain length and the packet crossover ratio. In Section 4, we propose a novel network-based SSID algorithm using packet crossover. In Section 5, we present and analyze the results of our well-designed network experiments. Conclusion and discussion of future research directions will be given in Section 6.

2. Preliminaries

In this section, we introduce some basic concepts and preliminary knowledge that can help readers understand our SSID algorithm, and the rationale of estimating a connection chain length by using crossover packets.

2.1. Definitions of Send and Echo Packets

Network traffic for data communication is the interactions between a client machine's requests and the corresponding server machine's responses. Packets for data communication can be modeled as Send, Echo, and Ack ones in this paper.

Assume that Host i is the sensor host in the connection chain in Figure 1 above. A Send packet can be either a TCP packet sent from Host i-1 to Host i with the TCP.Flag.PSH flag bit being true, or a TCP packet sent from Host i to Host i+1 with the TCP.Flag.PSH flag bit being true. Similarly, an Echo packet can be either a TCP packet sent from Host i back to Host i-1 with the TCP.Flag.PSH flag bit being true, or a TCP packet sent from Host i+1 back to Host i with the TCP.Flag.PSH flag bit being true.

In order to help readers better understand the concepts of Send packet and Echo packets, an example will be used to explain what a matched pair of Send packet and Echo packets is. For instance, in a Linux machine, we type a command like "ls" on a terminal. This command "ls" is assumed to be sent to the victim host in two different packets, one for the letter "l" and the other one for the letter "s". According to the definitions above, both packets are Send ones. After a user types the letter "l" on Host 0 (see Figure 1), the corresponding Send packet will be sent to the victim host (Host N). After this packet holding "l" is received and processed by Host N, Host N will send an Echo packet to Host

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 August 2023 doi:10.20944/preprints202308.0200.v1

0. Once Host 0 receives this Echo packet, the letter "l" will display on its screen. Within such a transaction, the Send holding "l" and the Echo holding "l" form a matched pair. For the other Send packet holding "s", the letter "s" will display on the screen of Host 0 once it receives the corresponding Echo packet holding "s". The RTT of a matched pair of Send/Echo packets represent the length of a connection chain, the number of connections in the chain.

2.2. RTT Distribution

It is non-trivial to match a Send packet with its corresponding Echo, as multiple Send packets may be echoed by single Echo packet, or a Send packet may be echoed by multiple Echo packets. In this paper, we compute the RTT between a matched pair of Send and Echo packets by matching the corresponding Send/Echo packets. We do not need to match a Send with all of its possible Echo packets. For simplicity, we match a Send with its first Echo packet and then computer their RTT [9].

The RTT of a packet is represented by the summation of four different types of delays. They are respectively the propagation delay, transmission delay, queuing delay, and processing delay. It is well-known that a network connection delay can be modelled as a queue. Let *T(t)* be the RTT of a network connection. Then we have

$$T(t) = T_0 + \Delta T(t), \tag{1}$$

where T_0 is a constant, and $\Delta T(t)$ is a variation of the packet RTT. The value of T_0 is mainly determined by the propagation delay, and the value of $\Delta T(t)$ is mainly determined by the queuing delay [21]. If the /M/M/1 queuing model is employed to compute the RTT queue [14]. Then the distribution of $\Delta T(t)$ can be obtained as follows:

$$\begin{aligned} P(\Delta T > x) &= \lim_{t\to\infty} P(\Delta T(t) > x) \\ &= e^{(-\gamma_i x)}, \quad \gamma_i = \mu_i(1-\rho_i) \end{aligned} \tag{2}$$

where μ_i is the traffic rate on the network link *i*, and ρ_i is the corresponding utilization factor of this link.

The above Equation (2) shows that the variance of RTTs follows an exponential distribution. Due to the cumulative delay of all the stepping-stone hosts between Host 0 and Host N (see Figure 1) in the connection chain, it is very difficult to simulate the variance of RTTs as an exponential distribution. Discovered in a seminar work [14] by V. Paxson et al., the values of a connection chain's ΔRTT follows Poisson distribution [14,21]. This Poisson model is used for the values of a connection chain's ΔRTT in this paper.

2.3. Chaff Attack Definition

Today, intruders tend to launch cyberattacks with session manipulation techniques such as chaff perturbation. Chaff-perturbation is a hacking technique that intruders inject meaningless packets into a communication session in order to modify not only the packets' RTTs, but also the total number of packets within a certain period of time. Many existing SSID approaches can easily be defeated by chaff perturbation. Chaff attacking technique is widely used in the attacks such as man-in-the-middle, DoS, DDoS, or SSI attacks.

2.4. The Rationale of Network-based SSID Algorithms

Many legitimate applications use one stepping-stone host to access a remote server. For instance, when a Web browser sends a request to a remote Web server for some resources, the Web server might need to access a remote database server to prepare for the response message for that request from the browser. In such a case, the client browser remotely gains access to the database server through the Web server. Thus, the Web server serves as a stepping-stone host in such a scenario. Also, cloud-computing applications might use one stepping-stone host to access a remote server in the cloud.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 August 2023 doi:10.20944/preprints202308.0200.v1

Clearly, the more hosts involved in accessing a remote server, the slower the network traffic is. Some research shows that legitimate applications rarely use two or more stepping-stone hosts to access a remote server. It is unnecessary to access a remote server indirectly via two or more stepping-stones as doing so will produce a great number of unnecessary network packets. Such an access to a remote server would be ineffective and inefficiency. Thus, we can conclude that if two or more stepping-stones are used to gain access to a remote server, it is most likely the session is a malicious intrusion. Therefore, we can estimate the number of connections in a chain to detect SSI.

2.5. Packet Crossover

Packet crossover occurs when an Echo packet of a previous Send packet meets a new Send packet along the connection chain between Host 0 and Host N (see Figure 1). Let us use the example in Figure 2 to explain the meaning of packet crossover. In Figure 2, the length of the connection chain is three, Host 1 (client) serves as the attacker host, Host 4 (server) serves as the victim host, and Host 2 and Host 3 serve as the stepping-stone hosts. The red packets S1, S2 and S3 are the Send ones, and the green packets E1, E2, and E3 are the Echo ones. Assume that Host 1 is used as the sensor, then all the Send and Echo packets between the connection of Host 1 and Host 2 will be captured and analyzed. In such a scenario, the order of these six packets is in turn S1, S2, E1, S3, E2, and E3 based on their time stamps. Clearly, packet crossover occurs twice in this case. If Host 2 is selected as the sensor host, then all the Send and Echo packets between the connection from Host 2 to Host 3 will be captured and analyzed. The order of these six packets captured at Host 2 is S1, E1, S2, S3, E2, and E3, based on their time stamps. It is easy to see that packet crossover occurs only once in this case.

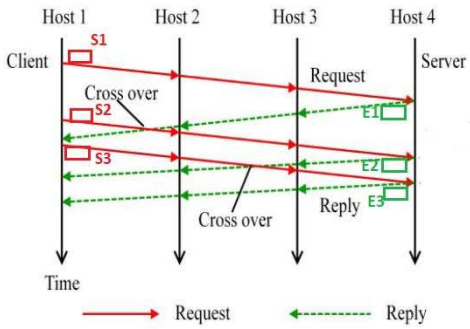

Figure 2. A sample of packet crossover in a connection chain of four hosts.

3. Relationship between the downstream sub-chain length and packet crossover ratio

In this section, we give a proposition that asserts the relationship between the downstream sub-chain length and the packet crossover ratio observed at the sensor host. The following proposition was proved in our prior work [20].

Proposition 1: For a given connection chain, the length of the downstream sub-chain from the sensor to the target strictly increases with the packet crossover ratio observed at the sensor host.

In this paper, we verified through well-designed network experiments in Section 6 that Proposition 1 above is true when the network traffic has no chaffed meaningless packets. More importantly, the proposition remains true when intruders' chaff-perturbation is present up to 50% chaff rate. We collected 10 datasets in total, and the error rate remains 0% when the network traffic is chaffed meaningless packets with a chaff rate of 10%, 20%, 30%, 40%, or 50%, respectively. This proposition will play an important role in designing and analyzing our SSID algorithm to be proposed in Section 4 below.

4. SSID Algorithm Design

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 August 2023 doi:10.20944/preprints202308.0200.v1

In this section, we develop a novel network-based SSID algorithm for network traffic in any LAN or WAN with or without chaffed meaningless packets by estimating the length of a downstream sub-chain. Our SSID algorithm works effectively to detect SSI in any LAN or WAN as well as resists to intruders' chaff-perturbation. The design and analysis of our SSID algorithm are based on Proposition 1 stated above.

Based on our discussion in Section 2.4, it is most likely that SSI is present if the length of a downstream sub-chain is at least two, which makes the length of the whole connection chain be at least three as the upstream sub-chain length is at least one.

Pick any host in the network as the sensor host S1. Our proposed SSID algorithm determines whether the sensor S1 is used by an intruder for SSI. The ideas of our proposed SSID algorithm via estimating the length of a downstream sub-chain are listed below:

(1) Set up a connection chain A→S1→S2→V of length 3 with the host S1 as the sensor, where the hosts A, S2, and V serve as the attacker, another stepping-stone, and the victim, respectively. The length of the downstream sub-chain from S1 to V is two.
(2) Some standard Linux commands (such as ls, dir, mkdir, etc.) are entered into a terminal in the attacker host A for a couple of minutes, and at the same time all the packets are captured at the sensor S1 from the connection S1→S2 in the chain. Totally, 10 datasets will be captured. Then we use the Packet Crossover Ratio algorithm (Algorithm 1 of [20]) to calculate the packet crossover ratio for each dataset of the above captured packets.
(3) Calculate the intrusion threshold crossover ratio which is the average packet crossover ratio among the 10 captured datasets at Step 2.
(4) To perform SSID, at the same time, we also use host S1 as the sensor and observe one of its outgoing links. We then determine whether this outgoing link from the sensor S1 is used by an intruder for a malicious SSI. We capture 10 datasets at the sensor S1 from this outgoing connection and calculate the average packet crossover ratio over all the 10 captured datasets using the Packet Crossover Ratio algorithm (Algorithm 1 of [20]).
(5) If the average packet crossover ratio obtained at Step 4 is greater than or equal to the intrusion threshold crossover ratio obtained at Step 3, it is most likely that this outgoing link is used by a hacker for malicious SSI.
(6) Repeat Step 4 for every outgoing link from the sensor S1 (except for the connection S1→S2 in the chain created in Step 2) to see whether it is used by a hacker for malicious SSI.

At Step 5, if the obtained average packet crossover ratio is less than the intrusion threshold crossover ratio obtained at Step 3, we can conclude that the length of the downstream sub-chain is less than two, according to Proposition 1. However, we are unable to tell whether there is an intrusion or not in such a case as we do not know the length of the upstream sub-chain. It requires further analysis on the upstream sub-chain length.

5. Network Experimental Results and Analysis

In this section, we present and analyze the results of our network experiments that were well-designed to verify the following:

(1) The correctness of Proposition 1 stated in Section 3;
(2) The correctness of our proposed SSID algorithm stated in Section 4.

To set up the network environment for the experiment, we used two local hosts and six Amazon AWS servers distributed in different regions, all of which are running Ubuntu operating systems. A long chain was established by using Secure Shell (SSH) to connect to all the machines in the chain from the attacker machine H1 to the victim target H8 (refer to Figure 3). Host H1 is a local PC at Columbus State University in Georgia, USA with a private IP 168.27.2.101. From H1, we accessed remotely another local host H2 which is also located at Columbus State University in Georgia, USA with a private IP 168.27.2.103. Then the chain is extended to access remotely an AWS server H3 located in Virginia, USA with a public IP 54.175.200.189, by using H2 as a stepping-stone. The chain is extended again to access remotely another AWS server H4 located in London, England with a public IP 35.178.87.47, by using H3 as a stepping-stone. The chain is extended again to remotely access

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 August 2023 doi:10.20944/preprints202308.0200.v1

another AWS server H5 located in Virginia, USA with a public IP 3.87.217.13, by using H4 as a stepping-stone. Similarly, the connection chain is extended again to access remotely another AWS server H6 located in Tokyo, Japan with a public IP 54.65.202.87, by using H5 as a stepping-stone. The chain is extended again to access remotely another AWS server H7 located in Paris, France with a public IP 15.188.87.227, by using H6 as a stepping-stone. Finally, the connection chain is extended again to access remotely the victim target H8 (an AWS server) located in Virginia, USA with a public IP 54.86.84.197, by using H7 as a stepping-stone. This completes the setup of the connection chain.

The network packets were captured using TCPdump from the downstream connection at every selected sensor host in the chain. For example, when Host 3 is chosen as the sensor, then the packets are captured from the connection between Hosts 3 and 4.

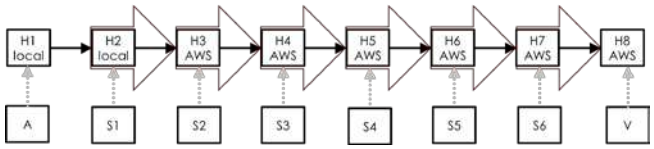

Figure 3. A chain of 7 connections with host H3 thru host H7 serving as sensors respectively, used to capture network traffic from the downstream connection.

Our 1st experiment was conducted for network traffic without any chaffed meaningless packets. At every sensor host, as soon as TCPdump is ready to capture network packets, some standard Linux commands (such as ps, ls, dir, etc.) are entered into a terminal in the attacker machine H1 for a couple of minutes and all the packets from the indicated connection were captured at each of the 5 sensor hosts (H3 thru H7) in the chain. Totally, 10 datasets were captured at each of the 5 sensor hosts (H3 thru H7). Then we use our Packet Crossover Ratio algorithm [20] to calculate the packet crossover ratio based on the captured packets at each specific sensor host. Finally, we compute the average packet crossover ratio among the 10 datasets.

In Table 1, column 1 (# of Conn) represents the number of connections in the downstream sub-chain, and "DS-1" stands for a data set #1. Columns 2 through 11 show the packet crossover ratios calculated for a specified dataset from DS-1 to DS-10, with a given number of connections specified in column 1. For each of the 10 datasets (represented in Table 1 from column 2 through column 11, respectively), the length of a downstream sub-chain strictly increases with the packet crossover ratio observed at the sensor host. Therefore, Proposition 1 is true for network traffic without chaff. Therefore, our experimental results completely support the statement of Proposition which is 100% correct for each of the 10 datasets. That is, the error rate is 0%.

In Table 1, the last column is the average packet crossover ratios among the 10 datasets. In this experiment, the intrusion threshold crossover ratio is **0.52**, which is the average packet crossover ratio derived from a downstream sub-chain of 2 connections over the 10 datasets. According to our proposed SSID algorithm, for a given outgoing link of the sensor host, if the average packet crossover ratio obtained at Step 4 of the algorithm over 10 datasets is not less than **0.52**, then it is highly suspicious that this outgoing link is used by a hacker for malicious SSI.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 August 2023 doi:10.20944/preprints202308.0200.v1

Table 1. Packet crossover ratios without chaff.

#ofConn	DS-1	DS-2	DS-3	DS-4	DS-5	DS-6	DS-7	DS-8	DS-9	DS-10	AVG
1	0.01	0.05	0.04	0.13	0.01	0.03	0.03	0.03	0.01	0.04	**0.04**
2	0.47	0.53	0.45	0.79	0.46	0.46	0.51	0.44	0.51	0.59	**0.52**
3	0.69	0.77	0.75	1.16	0.72	0.66	0.75	0.68	0.75	0.86	**0.78**
4	0.78	0.94	0.92	1.36	0.95	0.78	0.91	0.76	0.87	1.07	**0.93**
5	0.87	1.06	1.03	1.53	1.06	0.86	1.06	0.86	0.99	1.22	**1.05**

Our 2nd experiment was conducted for the network traffic with meaningless packets chaffed at a rate of 10%. We performed all the steps as we did for the 1st experiment above for network traffic without chaffed packets. Similarly, we use our Packet Crossover Ratio algorithm to calculate the packet crossover ratio based on the captured packets with 10% chaff rate at each specific sensor host. The results we obtained are very similar to those we obtained at the 1st experiment above.

In Table 2, the meanings of the columns are the same as in Table 1. For each of the 10 datasets (represented in Table 2 from column 2 through column 11, respectively), the length of a downstream sub-chain strictly increases with the packet crossover ratio observed at the sensor host. Therefore, our experimental results also completely support the statement of Proposition 1 for network traffic with a 10% chaff rate.

In Table 2, the last column is also the average packet crossover ratios among the 10 datasets. In this experiment, the intrusion threshold crossover ratio is **8.72**, which is the average packet crossover ratio derived from a downstream sub-chain of 2 connections over the 10 datasets. According to our proposed SSID algorithm, for a given outgoing link of the sensor host, if the average packet crossover ratio obtained at Step 4 of the algorithm is not less than **8.72**, then it is highly suspicious that this outgoing link is used by a hacker for malicious SSI.

Table 2. Packet crossover ratios with a 10% chaff rate.

#ofConn	DS-1	DS-2	DS-3	DS-4	DS-5	DS-6	DS-7	DS-8	DS-9	DS-10	AVG
1	11.46	6.57	7.66	8.41	8.13	6.68	7.89	6.08	6.46	8.69	**7.80**
2	11.89	7.02	8.03	9.11	8.55	7.11	8.37	6.43	6.92	9.26	**8.27**
3	12.20	7.27	8.33	9.43	8.87	7.34	8.64	6.7	7.14	9.51	**8.54**
4	12.23	7.46	8.52	9.68	9.06	7.43	8.79	6.86	7.33	9.76	**8.71**
5	12.32	7.59	8.63	9.83	9.19	7.49	8.94	6.87	7.44	9.9	**8.82**

Similarly, our 3rd through 6th experiments were conducted for the network traffic with meaningless packets chaffed at a rate of 20%, 30%, 40%, and 50%, respectively (corresponding results shown in Tables 3–6, respectively). For each of these experiments, we performed all the steps as we did for the 1st experiment above. Then we use our Packet Crossover Ratio algorithm to calculate the packet crossover ratio based on the captured packets from the network traffic with a specific chaff rate. The results we obtained are very similar to those we obtained at the 1st experiment above. In Tables 3 through 6, the meanings of the columns are the same as in Table 1. For each of the 10 datasets in every of these experiments from the 3rd to the 6th (corresponding results shown in Tables 3–6, respectively), the length of a downstream sub-chain strictly increases with the packet crossover ratio observed at the sensor host. Our experimental results also completely support this statement which is 100% true for each of the 10 datasets for network traffic with a chaff rate of 20%, 30%, 40%, and 50%, respectively.

The last column of Tables 3–6 is also the average packet crossover ratios among the 10 datasets. In the 3rd through 6th experiments, the intrusion threshold crossover ratios are respectively, **15.43**, **21.84**, **27.75**, and **33.18**.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 August 2023 doi:10.20944/preprints202308.0200.v1

Table 3. Packet crossover ratios with a 20% chaff rate.

#ofConn	DS-1	DS-2	DS-3	DS-4	DS-5	DS-6	DS-7	DS-8	DS-9	DS-10	AVG
1	21.91	12.53	14.58	16.05	15.57	12.82	15.13	11.55	12.38	16.66	**14.92**
2	22.41	13.03	15.03	16.65	16.02	13.27	15.69	12.10	12.89	17.20	15.43
3	22.66	13.31	15.31	17.09	16.23	13.45	15.88	12.31	13.18	17.51	**15.69**
4	22.76	13.46	15.59	17.29	16.54	13.47	16.04	12.34	13.26	17.74	**15.85**
5	22.83	13.54	15.72	17.50	16.74	13.73	16.19	12.44	13.38	17.88	**16.00**

Table 4. Packet crossover ratios with a 30% chaff rate.

#ofConn	DS-1	DS-2	DS-3	DS-4	DS-5	DS-6	DS-7	DS-8	DS-9	DS-10	AVG
1	31.37	17.93	20.89	22.88	22.21	18.30	21.64	16.56	17.68	23.81	**21.33**
2	31.90	18.36	21.31	23.64	22.68	18.77	22.11	16.96	18.25	24.37	21.84
3	32.16	18.66	21.58	23.92	23.03	18.97	22.35	17.22	18.43	24.64	**22.10**
4	32.24	18.85	21.94	24.16	23.21	19.06	22.52	17.38	18.59	24.87	**22.28**
5	32.36	18.96	22.03	24.32	23.35	19.19	22.66	17.46	18.71	25.04	**22.41**

Table 5. Packet crossover ratios with a 40% chaff rate.

#ofConn	DS-1	DS-2	DS-3	DS-4	DS-5	DS-6	DS-7	DS-8	DS-9	DS-10	AVG
1	40.06	22.82	26.69	29.22	28.33	23.35	27.58	21.15	22.58	30.37	**27.22**
2	40.55	23.30	27.12	29.95	28.82	23.88	28.12	21.62	23.11	30.98	27.75
3	40.86	23.60	27.34	30.22	29.16	24.01	28.30	21.78	23.41	31.25	**27.99**
4	40.94	23.79	27.69	30.51	29.37	24.12	28.46	21.99	23.53	31.46	**28.19**
5	41.00	23.92	27.82	30.64	29.51	24.23	28.67	22.04	23.60	31.72	**28.32**

Table 6. Packet crossover ratios with a 50% chaff rate.

#ofConn	DS-1	DS-2	DS-3	DS-4	DS-5	DS-6	DS-7	DS-8	DS-9	DS-10	AVG
1	48.05	27.45	31.97	35.08	34.04	28.08	33.03	25.37	27.07	36.42	**32.66**
2	48.55	27.90	32.43	35.78	34.57	28.54	33.55	25.75	27.61	37.07	33.18
3	48.82	28.16	32.72	36.03	34.72	28.76	33.82	26.04	28.01	37.33	**33.44**
4	48.94	28.30	33.05	36.29	35.07	28.77	34.09	26.16	28.04	37.61	**33.63**
5	49.01	28.43	33.22	36.50	35.21	29.04	34.24	26.23	28.20	37.66	**33.77**

6. Conclusion and Future Work

In this paper, we developed a novel network-based SSID approach using packet crossover that is resistant to intruders' chaff manipulation. To the best of our knowledge, this paper is the first network-based SSID approach that resists intruders' session manipulation such as chaff perturbation. Since packet crossovers can be easily calculated, the SSID method proposed in this paper is easy to implement and efficient. Like prior network-based SSID approaches, our proposed SSID method also produces very low false-positive errors. According to our experiment results, our proposed SSID algorithm performs perfectly in resisting intruders' chaff-perturbation up to 50% chaff rate. The error rate is zero for any LAN or WAN network traffic with a chaff rate of at most 50%.

As for future research direction, one may develop SSID algorithms that are resistant to intruders' chaff manipulation if two or more hosts in a connection chain have chaffed meaningless packets by an intruder.

Author Contributions: Dr. L. Wang: SSID design, writing, analysis, supervision, and project administration; Dr. J. Yang: validation, analysis, investigation, supervision, and project administration; Mr. J. Kim: collect and analyze network packets; Dr. P.-J. Wang: supervision, help analyze the algorithm correction and validation. All authors have read and agreed to the current version of the manuscript.

Funding: This work of Drs. Lixin Wang and Jianhua Yang is supported by the National Security Agency NCAE-C Research Grant (H98230-20-1-0293) with Columbus State University, Georgia, USA.

Data Availability Statement: N/A.

Acknowledgments: N/A.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. L. Wang, J. Yang, X. Xu, and P.-J. Wan: "Mining Network Traffic with the k-Means Clustering Algorithm for Stepping-stone Intrusion Detection", *Wireless Communications and Mobile Computing*, vol. 2021, Article ID 6632671, 2021.
2. A. Blum, D. Song, And S. Venkataraman, "Detection of Interactive Stepping-Stones: Algorithms and Confidence Bounds", Proceedings of International Symposium on Recent Advance in Intrusion Detection (RAID), Sophia Antipolis, France, pp. 20-35, September 2004.
3. Bishop Mathew. "UNIX security: threats and solutions", Invited talk given at the 1995 system administration, networking, and security conference, Washington, DC, April 1995.
4. Bhattacherjee, Debopam. "Stepping-stone detection for tracing attack sources in Software-Defined Networks", Degree Project in Electrical Engineering, Stockholm, Sweden (2016).
5. D. Donoho, A. Flesia, U. Shankar, V. Paxson, J. Coit, and S. Staniford, "Multiscale stepping-stone detection: Detecting pairs of jittered interactive streams by exploiting maximum tolerable delay," in the 5th International Symposium on Recent Advances in Intrusion Detection, Lecture Notes in Computer Science, 2002.
6. Liu, Jinping, Wuxia Zhang, Zhaohui Tang, Yongfang Xie, Tianyu Ma, Jingjing Zhang, Guoyong Zhang, and Jean Paul Niyoyita. "Adaptive intrusion detection via GA-GOGMM-based pattern learning with fuzzy rough set-based attribute selection." *Expert Systems with Applications* 139 (2020): 112845.
7. J. Yang, S.-H. S. Huang, "A Real-Time Algorithm to Detect Long Connection Chains of Interactive Terminal Sessions," Proceedings of 3rd ACM International Conference on Information Security (Infosecu'04), Shanghai, China, November 2004, pp. 198-203.
8. J. Yang, S.–H. S. Huang, "Matching TCP Packets and Its Application to the Detection of Long Connection Chains," Proceedings of 19th IEEE International Conference on Advanced Information Networking and Applications (AINA 2005), Taipei, Taiwan, China, March 2005, pp. 1005-1010.
9. J. Yang, and S. S.-H. Huang, "Mining TCP/IP Packets to Detect Stepping-Stone Intrusion", *Journal of Computers and Security*, Elsevier Ltd., vol.26, December 2007, pp. 479-484.
10. Yang, J., Wang, L., Lesh, A., & Lockerbie, B. (2018). Manipulating network traffic to evade stepping-stone intrusion detection. *Internet of Things by Elsevier*, 3, 34-45. December 2018.
11. K. H. Yung, "Detecting Long Connecting Chains of Interactive Terminal Sessions," Proc. of International Symposium on Recent Advance in Intrusion Detection (RAID), Zurich, Switzerland, pp. 1-16, October 2002.
12. Phaal, P., Panchen, S., and McKee, N., "InMon Corporation's sFlow: A method for monitoring traffic in switched and routed networks". *RFC 3176*, IETF, September 2001.
13. S. Staniford-Chen, and L. T. Heberlein, "Holding Intruders Accountable on the Internet," Proc. IEEE Symposium on Security and Privacy, Oakland, CA, pp. 39-49, 1995.
14. V. Paxson, S. Floyd, "Wide-area Traffic: The Failure of Poisson Modeling", *IEEE/ACM Transactions on Networking*, (3) 1995, pp. 226-244.
15. L. Wang and J. Yang. "A research survey in stepping-stone intrusion detection." *EURASIP Journal on Wireless Communications and Networking* 2018.1 (2018): 1-15.
16. Wang, X., & Reeves, D., "Robust correlation of encrypted attack traffic through stepping-stones by flow watermarking". *IEEE Transactions on Dependable and Secure Computing*, 8(3), pp. 434-449, 2011.
17. Chen, Y., and Wang, S., "A Novel Network Flow Watermark Embedding Model for Efficient Detection of Stepping-stone Intrusion Based on Entropy", In Proceedings of the International Conference on e-Learning, e-Business, Enterprise Information Systems, and e-Government (EEE), WorldComp 2016.
18. Y. Zhang, and V. Paxson, "Detecting Stepping-Stones," In Proceedings of the 9th USENIX Security Symposium, Denver, CO, pp. 67-81, August 2000.
19. L. Wang, J. Yang, and A. Lee: An Effective Approach for Stepping-Stone Intrusion Detection Using Packet Crossover, the 23rd World Conference on Information Security Applications (WISA), Aug. 24-26, 2022.
20. Wang, L.; Yang, J.; Lee, A., and Wan, P.-J., Stepping-Stone Intrusion Detection via Estimating Numbers of Upstream and Downstream Connections using Packet Crossover, *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications* (JoWUA), Vol. 13, No. 4 (Dec. 2022).
21. Li, Q.; Mills, D.L. On the Long-range Dependence of Packet Round-trip Delays on the Internet. In Proceedings of International
22. Conference on Communications (ICC'98), Atlanta, GA, USA, 7–11 June 1998; Volume 1, pp. 1185–1192.

 doi:10.20944/preprints202308.0200.v1