Stepping-Stone Intrusion Detection via Estimating Numbers of Upstream and Downstream Connections using Packet Crossover*

Jianhua Yang[1], Lixin Wang[1]†, Austin Lee[1], and Peng-Jun Wan[2]

[1]Columbus State University, Columbus, GA 31907, USA
{yang_jianhua, wang_lixin, lee_austin}@columbusstate.edu

[2]Illinois Institute of Technology, Chicago, IL 60616, USA
wan@cs.iit.edu

Received: September 28, 2022; Accepted: December 4, 2022; Published: December 31, 2022

Abstract

An effective approach to detect stepping-stone intrusion (SSI) is to estimate the length of a connection chain. This type of detection method is referred to as the network-based SSI detection (SSID). Previously known network-based approaches for SSID either do not work effectively in the Internet environment, or are inefficient as they require a large number of packets to be captured and analyzed, or have limited performance as the length of a connection chain must be predetermined. None of these existing methods to detect SSI can be used to estimate the length of upstream connection sub-chain, which has been a long-standing and challenging open problem in this research area. In this paper, we develop effective network-based methods for SSID using packet crossover that can be used to estimate the length of a downstream sub-chain as well as that of an upstream sub-chain. Since the number of packet crossovers can be easily calculated, our proposed algorithms for SSID are easy to use and implement. To the best of our knowledge, this is the first work that can effectively estimate the length of the whole connection chain, including the upstream sub-chain. Rigorous technical proofs and well-designed network experiments are provided to verify the correctness and effectiveness of our proposed algorithms for SSID.

Keywords: Stepping-stone intrusion, upstream intrusion detection, connection chain, upstream sub-chain, downstream sub-chain, packet crossover

1 Introduction

Cyber attacks on remote target systems are often launched through compromised hosts, referred to as stepping-stones, aiming to reduce the chance of being detected [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15].

Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, 13(4):24-39, Dec. 2022
DOI:*10.58346/JOWUA.2022.I4.002*

*Part of this work was presented in the 23rd World Conference on Information Security Applications (WISA), 2022 [1].

†Corresponding author: Lixin Wang. Dr. Lixin Wang is with Columbus State University, Columbus, GA 31907, USA. E-mail: wang_lixn@columbusstate.edu.

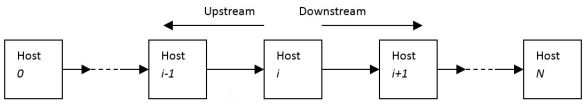

Figure 1: A sample of a connection chain.

Using stepping-stone intrusion (SSI), an attacker uses a chain of stepping-stones on the Internet as relay hosts and remotely login these hosts by using programs such as SSH, rlogin or telnet. An intruder sends attacking commands at his/her local machine and the packets are relayed via multiple stepping-stone hosts until they reach the remote target system under attack.

With SSI, an attacker builds a connection chain (see Fig. 1) and uses SSH to remotely login to these stepping-stones in turn and then sends attacking commands. In Fig. 1, Host 0 is the machine operated by the attacker and used to login to the stepping-stone hosts Host 1, Host 2, . . ., Host i-1, Host i, Host i+1, . . ., and Host N -1. Host N is the remote target system that is being attacked. For SSI detection (SSID), any stepping-stone host (other than the attacker host) could be used as the sensor host in which a detecting program or packet sniffer, such as TCPdump or Wireshark, is installed and used to capture packets. In this figure, Host i is assumed to be the detecting sensor host with a detecting program installed. An incoming connection to the sensor Host i is the connection from Host i-1 to Host i, and an outgoing connection from Host i is the connection from Host i to Host i+1. In terms of the sensor Host i, the part of the connection chain from the attacker Host 0 to the sensor Host i is called an upstream sub-chain, and the part of the connection chain from the sensor Host i to the target Host N is called a downstream sub-chain. If there is at least one relayed pair between all the incoming connections and all the outgoing connections of the sensor Host i, then it is most likely that Host i is used as a stepping-stone and the session is manipulated by a hacker. The purpose of SSID is to determine whether the sensor Host i is used as a stepping-stone host by an intruder. One type of detection method for SSID is to compare all the outgoing connections with all the incoming connections of the same host to see if there exists a relayed pair. This type of detection approach is called the host-based SSID [3, 5, 6, 9, 10, 11, 12, 13, 14, 15]. This type of detection method focuses on a single host (the sensor) for the detection. Because legitimate applications, such as Web applications and cloud computing applications, often utilize stepping-stone hosts to access a remote server. Therefore, this type of methods may produce high false-positive errors. Also, this type of SSID is not effective when intruders use time-jittering and/or chaff permutation for session manipulation.

To reduce false-positive errors for SSID, another type of approach was proposed to estimate the length of a connection chain, which is referred to as the network-based detection approaches [2, 16, 17, 18]. That is, this type of method focuses on estimating the number of connections from an intruder host to the remote target host in the chain (as shown in Figure 1). This number is called the length of a connection chain. If there exist three or more connections in a connection chain from the attacker host to the target, it means that the attacker machine tries to access a remote target system via two or more relayed stepping-stones. Clearly, the more hosts used in an interactive session to gain access to a remote server, the slower the network communication. It is well-known that legitimate applications

rarely utilize two or more relayed hosts as stepping-stones to access a remote server. Therefore, it is highly suspicious that it is a malicious intrusion if a remote server is accessed by a user via two or more relayed stepping-stones.

Next, we provide a literature review with a focus on network-based approaches for SSID that estimate the length of a connection chain. In 2002, Yung et al. [18] proposed the first network-based approach for SSID. Yung method [18] estimates the length of a connection chain length by computing the ratio between the Send-Echo RTT and of Send-Ack RTT. A Send-Echo RTT represents the length of the connection chain from the sensor to the target. On the other hand, a Send-Ack RTT can reflect the length from the sensor to its adjacent host along the downstream connection sub-chain. The false-positive error produced by the Yung method developed in [18] is slightly reduces, but very limited. However, Yung method for SSID generates high false-negative errors due to the use of the acknowledgement packets. The issues of Yung method was discussed in [16], which is the 2nd network-based approach for SSID proposed by Yang et al. in 2004. The detection algorithm proposed in [16] uses the step-function approach to calculate the length of a connection chain in a local area network (LAN). An improvement of the approach in [16] over Yung one in [18] is that the authors of [16] changed the way to set up the connection chain so that every Send packet can be matched a corresponding Echo packet. In a LAN, the step-function method for SSID worked well and reduced both the false positive error and the false negative error, compared to Yung method proposed in [18]. However, a major drawback of the detection method proposed in [16] only worked well within a LAN. It did not work effectively in the Internet environment. For the context of the Internet, a conservative and greedy packet matching algorithm for SSID was proposed in [19] by Yang et al. But only very few packets can be matched using this approach and thus it did not work effectively.

In 2007, Yang et al. [17] developed the clustering and partitioning data mining algorithm for SSID. The authors of [17] used a clustering and partitioning data mining method to calculate the RTTs of Send packets. The prior network-based approaches only compare one Echo packet with a Send packet at a time for matching the Send and corresponding Echo packets. [17] made the matching process much more accurate by checking all the possible Echo packets for each Send packet. In [17] utilized the maximum-minimum distance clustering algorithm (MMD) to compute the packets RTTs. The number of clusters generated by the MMD data mining algorithm reflects the number of connections in the chain. However, a major drawback of this approach is that a huge number of packets must be captured and analyzed, which makes this detection method not efficient in terms of processing time of captured packets.

A recent work [2] addressed the issue of [17] and developed an SSID method via mining network traffic through the k-Means clustering algorithm. This k-Means based approach proposed in [2] does not need to capture and analyze a large number of packets, and thus it is more efficient than MMD based method developed in [17]. However, a major drawback of the k-Means clustering based detection approach requires that the length of a connection chain must be pre-determined, which makes the performance of the SSID approach developed in [2] very limited. When large network fluctuations exist, the k-Means based detection method does not work effectively.

In this paper, we develop effective network-based methods for SSID using packet crossover that can estimate the length of a downstream sub-chain as well as that of an upstream sub-chain. Since the

number of packet crossovers can be easily calculated, our proposed algorithms for SSID are easy to use and implement. To the best of our knowledge, this is the first work that can effectively estimate the length of the whole connection chain, including the upstream sub-chain. Our proposed SSID algorithms using packet crossover do not have any pre-assumption about the length of a connection chain, nor require capturing and processing a large number of TCP packets. Therefore, our proposed SSID algorithms are efficient and easy to implement. Rigorous technical proofs and well-designed network experiments are provided to verify the correctness and effectiveness of our proposed algorithms for SSID.

The remaining of this paper is organized as follows. In Section 2, preliminary knowledge needed for proposing the SSID algorithms are given. In Section 3, the packet crossover ratio observed at the sensor host is calculated. In Section 4, we describe and prove a proposition that is needed for our SSID algorithm design and analysis. In Section 5, we estimate the length of a downstream sub-chain. In Section 6, we estimate the length of an upstream sub-chain. Network experiments to verify our proposed detection algorithms are presented in Section 7. Finally, we conclude this paper and give some future research directions in Section 8.

2 Preliminaries

In this section, we introduce some basic concepts that are helpful to understand our detection algorithm for SSI, and the rationale of using crossover packets to estimate the length of a connection chain.

2.1 Definitions of Send/Echo Packets

In Fig. 1, Host i is assumed to be the detecting sensor. In the incoming connection to Host i, a Send packet is a TCP packet with the flag bit TCP.Flag.PSH set that is received at Host i and sent from Host i-1; an Echo packet is a TCP packet with the TCP.Flag.PSH flag bit set that is received at Host i-1 and sent from Host i. In the outgoing connection from Host i, a Send packet is a TCP packet with the TCP.Flag.PSH flag bit set that is received at Host i+1 and sent from Host i; an Echo packet is a TCP packet with the TCP.Flag.PSH flag bit set that is received at Host i and sent from Host i+1.

Let us illustrate which Send packet and Echo packet are a matched pair using an example. When a command is typed on a terminal window of a Linux system, such as p̈s, we assume that the command p̈sïs delivered to the remote server in two separate packets: one holding p̈änd the other holding s̈. Both of them are Send packets. When the letter p̈ïs typed, the packet holding p̈ẅill be delivered to the remote server. Once this packet is received and processed at the server, an Echo packet is sent back to the sending host, p̈ẅill be displayed on the command line of the useŕs machine. In such a scenario, the Send packet p̈änd the Echo packet p̈äre a matched pair of packets. Similarly, a Send packet s̈änd its corresponding Echo packet s̈äre a matched pair.

The packet RTT can be calculated by using the timestamps of a matched pair of Send and Echo packets. The RTT of a matched packet pair reflects the length (number of connections) of a connection chain. Although the RTT obtained from the matched pair of the Send and Echo packets of p̈m̈aybe distinct with the RTT obtained from the matched pair of the Send and Echo packets of s̈, they should be

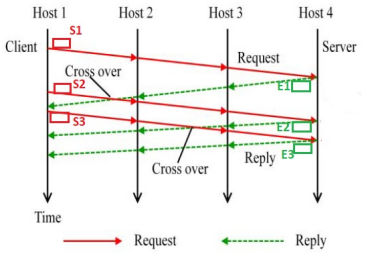

Figure 2: A sample of packet crossover in a connection chain of four hosts.

very close as they represent the length of the same chain with distinct times.

2.2 Packet Crossover

Packet crossover is a phenomenon in which a new Send (request) packet meets an Echo (reply) packet of a previous Send packet along the connection chain between a client host and a server host. For example, in Fig. 2, we have a connection chain starting from the client (Host 1), to Host 2, then to Host 3, and finally to the server (Host 4), where Host 2 and Host 3 are the stepping-stone hosts in this chain. The Send packets S1, S2 and S3 (marked red in the figure) are sent from the client host to the server, and their Echo packets are respectively, E1, E2, and E3 (marked green in Fig. 2). If packet crossover is observed at the client host (Host 1), all the Send and Echo packets from the connection between Host 1 and Host 2 are captured and analyzed. The sequence of these six packets is S1, S2, E1, S3, E2, and E3. Therefore, there are two occurrences of packet crossovers in this scenario, observed from the standing point of Host 1. However, if packet crossover is observed at Host 2, all the Send and Echo packets from the connection between Host 2 and Host 3 are captured and analyzed. The sequence of these six packets captured at Host 2 is S1, E1, S2, S3, E2, and E3. Therefore, there is only one occurrence of packet crossover in this scenario.

2.3 The Distribution of PacketsŔTTs in a Connection Chain

The length of a connection chain can be represented using the RTTs calculated from matched pairs of Send and Echo packets. It was proved in Yang et al. [16] that the number of connections of a connection chain equals the number of clusters generated by the packet RTTs that are computed from the connection chain.

The authors of [11] claimed and proved that the RTTs computed from a connection chain follow Poisson distribution. This discovery is quite useful. We can match the Send packets with their corresponding Echo packets, and then estimate the length of the connection chain. The results of a well-designed network experiment in [11] is shown in Figure 3 that the packet RTTs follow Poisson distribution. In this figure, the X-axis stands for the RTT values in micro-second, and the Y-axis stands for the chance of the occurrence of each RTT value. The RTT values were obtained from the TCP packets captured from a connection chain of length 4, with four connections and five hosts in total. In this experiment, the mean

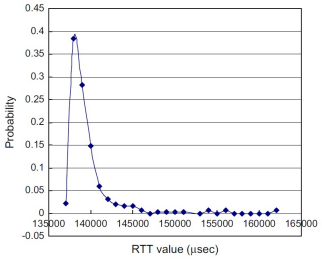

Figure 3: The Distribution of PacketsŔTTs for a Connection Chain.

of all the RTT values is μ = 138,500 (micro-second), and most of the RTT values are very close to the mean μ (see Fig. 3).

Given any random variable X obeying the Poisson distribution. Let μ and σ respectively denote its mean and standard deviation. According to a well-known inequality in Probability Theory, we have $|X - \mu| \leq 2\sigma$.

In other works, the absolute value of the difference between μ and X has an upper bounded of 2σ. That is, most values taken by the random variable X should be near its mean μ. Based on above discussions, the packet RTT values computed from a connection chain obey Poisson distribution. In other words, most of the packet RTT values computed from a connection chain should be near its mean value within a circle of radius 2σ.

2.4 The Rationale to Detect SSI Based on the Length of a Connection Chain

It is well-known that using one host as a stepping-stone is very common in legal applications. For example, one such a typical application is a Web application. Accessing a Web server from a client browser may result in accessing a database server through the Web server because most Web forms need data from its database server which normally resides in another different remote server. In this scenario, from the browser to the Web server, then to the database server, the Web server obviously plays a role as a stepping-stone host which is used legally.

However, it is well-known that it is quite rare to use three or relayed hosts as stepping-stones by legitimate applications. Clearly, the more hosts to pass through to access a remote target system, the slower the network traffic is. If there were no malicious behaviors to hide, it is neither wise nor necessary to access a target indirectly via three or more relayed hosts since the accessing process would generate a large amount of unnecessary traffic, resulting in inefficiency of the remote access. Therefore, it is reasonable to assume that it is highly suspicious to hide malicious activities if three or more relayed stepping-stone hosts are used to access a remote server. Thus, to detect stepping-stone intrusion, we can estimate the number of stepping-stone hosts used in a connection chain. That is, this detection method is to estimate the length of a connection chain. The longer a connection chain is, the more suspicious the session is.

To the best of our knowledge, it has been a long-standing open problem to estimate the length of

Algorithm 1: Calculate Packet Crossover Ratio

Input : a TXT file containing packet timestamp, packet type (Send or Echo), and index of Send or Echo

Output: Packet Crossover Ratio

```
sendIndex, echoIndex, crossoverCount = 0;
while more packets in data capture file
    if currentPacket is Acknowledgement
        discard packet;
        break;
    else if currentPacket is Echo
        if echoIndex less than sendIndex
            crossoverCount += sendIndex - echoIndex;
        echoIndex += 1;
    else if (currentPacket is Send)
        sendIndex += 1;
PacketCrossoverRatio = crossoverCount / (2 * echoIndex);
Print PacketCrossoverRatio;
```

the upstream sub-chain from the attacker to the sensor host for SSID, even though a few researchers made attempts to propose methods toward solving this problem. However, these methods for upstream detection only made a very limited progress. In this paper, we propose innovative algorithms to estimate the length of a downstream sub-chain from the sensor to the target, as well as the length of an upstream sub-chain from the attacker to the sensor. Since the length of an upstream (a downstream, respectively) sub-chain is at least one, if a downstream (an upstream, respectively) sub-chain has two or more relayed connections, then there are definitely three or more relayed connections used to access a remote server, and thus it is highly suspicious that there is an intrusion in such a case.

3 Calculate Packet Crossover Ratios

In this section, we present an algorithm to calculate packet crossover ratio that will be used to design innovative detection methods for SSI in later sections.

The input of this algorithm to compute packet crossover ratio is a TXT file containing collected packets with each packet including timestamp, packet type (Send or Echo), and index of Send or Echo. Such a TXT file is obtained from a raw PCAP files captured at the sensor host from a connection chain of a fixed length. The index of the first Send packet is initialized to one. The same is true for the first Echo packet. Every raw PCAP file is converted into a corresponding TXT file that contains three columns: packet timestamp, packet type, and index of Send or Echo packets. Our algorithm to compute the packet crossover ratio is described in Algorithm 1 (Calculate Packet Crossover Ratio).

4 A Proposition Needed to Design and Analyze the SSID Algorithm

In this section, we present and prove a proposition that asserts the relationship between the length of a downstream connection chain and the packet crossover ratio observed at the sensor host. This proposition will be verified by providing a rigorous technical proof in this section. It will be used to design and analyze our detection algorithms for SSID to be proposed in later sections.

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.*

Proof. First, let us use the above Fig. 2 as an example to present the main idea for this proof. If packet crossover is observed at the client host (Host 1), all the Send and Echo packets from the connection between Host 1 and Host 2 are captured and analyzed. Then, the sequence of these six packets is S1, S2, E1, S3, E2, and E3. Therefore, there are two occurrences of packet crossover in this scenario, observed from the standing point of Host 1. However, if packet crossover is observed at Host 2, all the Send and Echo packets from the connection between Host 2 and Host 3 are captured and analyzed. Then the sequence of these six packets captured at Host 2 is S1, E1, S2, S3, E2, and E3. Therefore, there is only one occurrence of packet crossover in this scenario. Thus, the number of occurrences of packet crossover is reduced by one if it is observed at Host 2. Similarly, if packet crossover is observed at Host 3, all the Send and Echo packets from the connection between Host 3 and Host 4 are captured and analyzed. Then the sequence of these six packets captured at Host 2 is S1, E1, S2, E2, S3, and E3. Therefore, there is NO packet crossover in this scenario. Again, the number of occurrences of packet crossover is reduced by one if it is observed at Host 3.

In general, when the sensor host of a connection chain is fixed, observation of packet crossover is always performed at the sensor host. That is, the network traffic of the first connection in the downstream sub-chain from the sensor host is captured and analyzed. Therefore, only the occurrences of packet crossover in the downstream sub-chain (from the sensor to the target) can be observed. Thus, the occurrences of packet crossover in the upstream sub-chain (from the attacker to the sensor) cannot be observed. This completes the proof of the proposition. □

For a given connection chain, since its total length (from the attacker host to the target) is fixed, when the length of its downstream sub-chain increases, the length of its upstream sub-chain decreases. According to the above Proposition 1, we have the following corollary:

Corollary 2. *For a given connection chain, the length of the upstream sub-chain from the attacker to the sensor host strictly decreases with the packet crossover ratio observed at the sensor host.*

5 Estimate the Number of Downstream Connections Using Packet Crossover

In this section, we present an effective algorithm to estimate the length of a downstream sub-chain (from the sensor to the target) using packet crossover. From our discussion in Section 2.4, it is most likely that

there is a malicious intrusion if the length of a downstream sub-chain is at least two, which makes the length of the whole connection chain is at least three as the length of the upstream sub-chain is at least one.

The idea of our proposed algorithm to estimate the length of a downstream sub-chain is listed below:

1. Calculate the intrusion threshold crossover ratio which is the average packet crossover ratio we obtained from a downstream connection chain of length two.

2. To perform SSID for a computer network, we pick a host in the network as the sensor and observe one of its outgoing links. We then determine whether this outgoing link from the sensor is used by a hacker for malicious intrusion.

3. We capture at least 10 datasets from this outgoing link from the sensor and calculate the average packet crossover ratio over all these datasets using Algorithm 1 described in Section III. If the obtained average packet crossover ratio is greater than the intrusion threshold crossover ratio, it is most likely that this outgoing link is used by a hacker for malicious intrusion.

4. Repeat Step 2 for every outgoing link from the sensor host to see whether it is used by a hacker for malicious intrusion.

Step 1 above can be done by setting up a connection chain with the length of its downstream sub-chain being two. Then we capture at least 10 datasets from the downstream chain from the sensor host and calculate the average packet crossover ratio over all the captured datasets using Algorithm 1 described in Section III. This intrusion threshold crossover ratio should be obtained before SSID is performed.

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

Correctness Analysis of Algorithm 1:

Now we analyze the correctness of the above proposed Algorithm 1 to estimate the length of a downstream sub-chain. According to our discussion in Section 2.4, since the length of an upstream sub-chain is at least one, if a downstream sub-chain has two or more relayed connections, then there are definitely three or more relayed connections used to access a remote server, and thus it is highly suspicious that there is an intrusion in such a case.

If the obtained average packet crossover ratio is greater than the intrusion threshold crossover ratio, according to Proposition 1, the length of the downstream sub-chain is at least two. Thus, it is most likely that this outgoing link from the sensor is used by a hacker for malicious intrusion.

6 Estimate the Number of Upstream Connections Using Packet Crossover

In this section, we propose an effective algorithm to estimate the length of an upstream sub-chain (from the attacker to the sensor host) using packet crossover. From our discussion in Section II.D, it is most

likely that there is a malicious intrusion if the length of an upstream sub-chain is at least two, which makes the length of the whole connection chain is at least three as the length of the downstream sub-chain is at least one.

The idea of our proposed algorithm to estimate the length of an upstream sub-chain is listed below:

1. Calculate the intrusion threshold crossover ratio which is the average packet crossover ratio we obtained from a downstream sub-chain of length two.

2. To perform SSID for a computer network, we pick a host in the network as the sensor and observe one of its incoming links. We then determine whether this incoming link to the sensor is used by a hacker for malicious intrusion.

3. We capture at least 10 datasets from this incoming link to the sensor and calculate the average packet crossover ratio over all these datasets using Algorithm 1 described in Section III. If the obtained average packet crossover ratio is less than the intrusion threshold crossover ratio, it is most likely that this incoming link is used by a hacker for malicious intrusion.

4. Repeat Step 2 for every incoming link to the sensor host to see whether it is used by a hacker for malicious intrusion.

At Step 3, if the obtained average packet crossover ratio is greater than the intrusion threshold crossover ratio, we can conclude that the length of the upstream sub-chain is less than two, according to Corollary 1. However, since we do not know an upper bound of the downstream sub-chain length, we are unable to tell whether there is an intrusion or not in such a case.

Correctness Analysis of Algorithm 1:

Now we analyze the correctness of the above proposed Algorithm 1 to estimate the length of an upstream sub-chain. Since the length of a downstream sub-chain is at least one, if an upstream sub-chain has two or more relayed connections, then there are definitely three or more relayed connections used to access a remote server, and thus it is highly suspicious that there is an intrusion in such a case.

If the obtained average packet crossover ratio is less than the intrusion threshold crossover ratio, according to Corollary 1, the length of the upstream sub-chain is at least two. Thus, it is most likely that this incoming link to the sensor is used by a hacker for malicious intrusion.

7 Network Experiments

In this section, we design network experiments to verify Proposition 1 described in Section IV. That is, we will verify that for a given connection chain, the length of a downstream sub-chain strictly increases with the packet crossover ratio observed at the sensor host.

To set up our experimental environment, we used two local hosts and six geographically dispersed Amazon AWS servers; all hosts in the experiment run the Ubuntu operating system. We created a long connection chain by using Secure Shell (SSH) to sequentially connect to each host in the connection

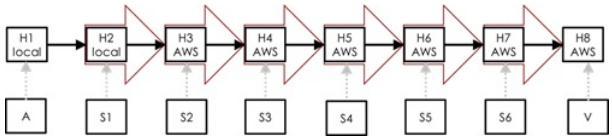

Figure 4: A connection chain of seven connections with Hosts 2 through 7 serving as sensors used to capture traffic from the downstream chain. Red arrows indicate the connection used for packet capture. For example, when H2 serves as the sensor, then we capture the network traffic between H2 and H3.

chain from the attacker host H1 to the victim host H8 (see Figure 4). In other words, a single terminal appearing on the local attacker host was used to create the entire connection chain by using sequential remote SSH access. From our local PC in Georgia, USA running Ubuntu with IP address 168.27.2.101, we remotely accessed the host H2 (the first stepping-stone host in the chain), located in Georgia, USA with IP address 168.27.2.103. We then extended the connection chain by using H2 as a stepping-stone to remotely access the host H3, located in Virginia, USA with public IP address 54.175.200.189. We then extended the connection chain again by using H3 as a stepping-stone to remotely access the host H4, located in London, England with public IP address 35.178.87.47. We then extended the connection chain by using H4 as a stepping-stone to remotely access the host H5, located in Virginia, USA with public IP address 3.87.217.13. We then extended the connection chain by using H5 as a stepping-stone to remotely access the host H6, located in Tokyo, Japan with public IP address 54.65.202.87. We then extended the connection chain by using H6 as a stepping-stone to remotely access the host H7 (the last stepping-stone in the chain), located in Paris, France with public IP address 15.188.87.227. We then extended the connection chain for the final time by using H7 as a stepping-stone to remotely access the victim host H8, located in Virginia, USA with public IP address 54.86.84.197; this final connection completed the chain of seven relayed connections. The tool TCPdump is used to capture the data for the downstream sub-chain on every selected sensor host in the connection chain. For example, at Host 2 we captured the traffic from the connection between Host 2 and Host 3; at Host 3 we captured the traffic from the connection between Host 3 and Host 4).

Once the TCPdump program ready to capture network traffic running at each sensor host, we entered some standard Linux commands (E.g., ls, cd, mkdir, etc.) for about three minutes into a terminal in the attacker host (H1) and captured all packets from the indicated connection in the chain. We captured ten datasets in total, with each data set comprising one file at each of the six sensor hosts in the connection chain. After capturing the data at each sensor, we ran our Packet Crossover Ratio algorithm to calculate the packet crossover ratio observed at a specific sensor host.

Clearly, we see that the length of a downstream sub-chain strictly increases with the ratio of packet crossover observed at the sensor host for a given connection chain. Our experimental results completely support this statement which is true for each of the datasets from dataset 1 through 10 (represented in Table 1 from column 2 through 10, respectively). For the average over these 10 datasets, the length of a downstream sub-chain increases strictly with the average packet crossover ratios. The error rate is 0% in our well-designed network experiment to verify the correctness of Proposition 1 proposed in Section

Conn#	DS-1	DS-2	DS-3	DS-4	DS-5	DS-6	DS-7	DS-8	DS-9	DS-10	AVG
1	0.0094	0.046	0.0362	0.1308	0.0147	0.0287	0.0303	0.0316	0.0074	0.0399	0.0375
2	0.4686	0.5331	0.4528	0.7945	0.4647	0.457	0.5106	0.4407	0.513	0.5895	0.5225
3	0.6948	0.7721	0.7453	1.1571	0.7206	0.6577	0.753	0.6818	0.7498	0.8567	0.7789
4	0.7823	0.9393	0.9169	1.3617	0.9456	0.776	0.9076	0.7648	0.868	1.073	0.9335
5	0.8688	1.0607	1.0266	1.5346	1.0603	0.8638	1.0621	0.8577	0.987	1.2176	1.0539
6	0.9129	1.0864	1.0549	1.5605	1.0882	0.8889	1.0864	0.876	1.0205	1.2452	1.0820

Table 1: The length of a downstream chain strictly increases with packet crossover ratio. D̈Sstands for a data set. AVG-Ratio is the average packet crossover ratio over the 10 datasets collected from a connection chain of the specified length.

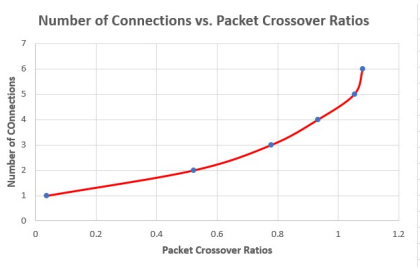

Figure 5: This figure illustrates the number of downstream connections strictly increases with the packet crossover ratio observed at the sensor host.

IV. In Table 1, column 1 represents the number of connections (conn#), columns 2 through 10 show the packet crossover ratio calculated from the specified dataset with a given number of connections, and the last column represents the average packet crossover ratio (AVG) over the 10 datasets. In the last column, the average packet crossover ratio 0.0375 corresponding to the number of downstream connections equal to 1, and the average packet crossover ratio 1.08199 corresponding to the number of downstream connections equal to 6. In this experiment, the intrusion threshold crossover ratio is 0.52245, which is the average packet crossover ratio derived from a downstream sub-chain of two connections over the 10 datasets.

Figure 5 clearly shows the positive relationship between the number of downstream connections in a chain and the packet crossover ratio observed at the sensor host.

Table 2 gives a comparison of all Network-based SSID (**NSSID**) methods, including the one proposed in this paper. This table highlights the novelty and contribution of this paper.

8 Conclusion

In this paper, we developed effective network-based methods for SSID by observing the number of packet crossovers that can be used to estimate the length of a downstream sub-chain as well as that of an upstream sub-chain. Since the number of packet crossovers can be easily calculated, our proposed

NSSID Method	**Year Proposed**	**Advantage**	**Drawback**
[18]	2002	The 1st NSSID method to estimate the length of a connection chain	High false-negative errors
[16]	2004	This method works well in local area networks	Does not work effectively in the Internet environment
[17]	2007	This method works well in the Internet environment	A huge number of packets must be captured and analyzed, which makes this method not efficient
[2]	2021	This method is efficient as it does not need to capture and analyze a large number of packets	It requires that the length of a connection chain must be predetermined, which makes the performance of this approach very limited
This paper	2022	It is the first work that can effectively estimate the length of the whole connection chain, including the upstream sub-chain. It is efficient and easy to implement.	

Table 2: Comparison of All Network-based SSID (**NSSID**) Methods.

algorithms for SSID are easy to implement. This is the first work that can estimate the length of the upstream sub-chain. Rigorous technical proofs and well-designed network experiments are provided to verify the correctness and effectiveness of our proposed algorithms for SSID. Previously known network-based SSID approaches are either not effective in the context of the Internet, or not efficient since a large number of packets must be processed, or with very limited performance as the length of a connection chain must be pre-determined.

As for future research directions, one may analyze the number of packet crossovers, and use it to match the incoming connections with the outgoing connections from the sensor host, and then determine whether the sensor host is used as a stepping-stone.

Acknowledgement

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.

References

[1] L. Wang, J. Yang, and A. Lee. An effective approach for stepping-stone intrusion detection using packet crossover. In *Proc. of the 23rd World Conference on Information Security Applications (WISA'22), Jeju,*

South Korea, pages 1–11. CISC, August 2022.

[2] 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*, 2021:1–9, March 2021.

[3] A. Blum, D. Song, , and S. Venkataraman. Detection of interactive stepping-stones: Algorithms and confidence bounds. In *Proc. of the 7th International Symposium on Recent Advance in Intrusion Detection (RAID'04), Sophia Antipolis, France*, volume 3224 of *Lecture Notes in Computer Science*, pages 258–277. Springer, Berlin, Heidelberg, September 2004.

[4] B. Mathew. Unix security: Threats and solutions. In *Proc. of the 1995 System Administration, Networking, and Security Conference (SANS'95), Washington, DC*. USENIX Association, April 1995.

[5] D. Bhattacherjee. *Stepping-stone detection for tracing attack sources in Software-Defined Networks*. PhD thesis, KTH, School of Information and Communication Technology (ICT), July 2016.

[6] 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 *Proc. of the 5th International Symposium on Recent Advances in Intrusion Detection (RAID'02), Zurich, Switzerland*, volume 2516 of *Lecture Notes in Computer Science*, pages 17–35. Springer, Berlin, Heidelberg, October 2002.

[7] J. Liu, W. Zhang, Z. Tang, Y. Xie, T. Ma, J. Zhang, G. Zhang, and J. P. Niyoyita. Adaptive intrusion detection via ga-gogmm-based pattern learning with fuzzy rough set-based attribute selection. *Expert Systems with Applications*, 139:1–17, January 2020.

[8] J. Yang, L. Wang, A. Lesh, and B. Lockerbie. Manipulating network traffic to evade stepping-stone intrusion detection. *Internet of Things*, 3:34–45, December 2018.

[9] P. Phaal, S. Panchen, and N. McKee. Inmon corporation's sflow: A method for monitoring traffic in switched and routed networks. RFC3176, September 2001. `https://www.rfc-editor.org/rfc/rfc3176` [Online; Accessed on December 15, 2022].

[10] S. Staniford-Chen and L. T. Heberlein. Holding intruders accountable on the internet. In *Proc. of the 1995 IEEE Symposium on Security and Privacy (SP'95), Oakland, CA, USA*, pages 39–49. IEEE, May 1995.

[11] V. Paxson and S. Floyd. Wide-area traffic: The failure of poisson modeling. *IEEE/ACM Transactions on Networking*, 3(3):226–244, June 1995.

[12] L. Wang and J. Yang. A research survey in stepping-stone intrusion detection. *EURASIP Journal on Wireless Communications and Networking*, 276(2018):1–15, December 2018.

[13] X. Wang and D. Reeves. Robust correlation of encrypted attack traffic through stepping-stones by flow watermarking. *IEEE Transactions on Dependable and Secure Computing*, 8(3):434–449, 2011.

[14] Y. Chen and S. Wang. 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'16), Las Vegas, NV, USA*, pages 1–6. CSREA Press, 2016.

[15] Y. Zhang and V. Paxson. Detecting stepping-stones. In *Proc. of the 9th USENIX Security Symposium (USENIX'00), Denver, CO, USA*, pages 67–81. USENIX Association, August 2000.

[16] J. Yang and S.-H. S. Huang. A real-time algorithm to detect long connection chains of interactive terminal sessions. In *Proc. of the 3rd ACM International Conference on Information Security (InfoSecu'04), Shanghai, China*, pages 198–203. ACM, November 2004.

[17] J. Yang and S.-H. S. Huang. Mining tcp/ip packets to detect stepping-stone intrusion. *Journal of Computers and Security*, 26(7-8):479–484, December 2007.

[18] K. H. Yung. Detecting long connecting chains of interactive terminal sessions. In *Proc. of the International Symposium on Recent Advance in Intrusion Detection (RAID'02), Zurich, Switzerland*, volume 2516 of *Lecture Notes in Computer Science*, pages 1–16. Springer, Berlin, Heidelberg, October 2002.

[19] J. Yang and S.-H. S. Huang. Matching tcp packets and its application to the detection of long connection chains. In *Proc. of 19th IEEE International Conference on Advanced Information Networking and Applications (AINA'05), Taipei, Taiwan*, pages 1005–1010. IEEE, March 2005.

Author Biography

Jianhua Yang is currently working at TSYS School of Computer Science, Columbus State University (CSU), Columbus, GA, USA as a Full Professor. Before joining CSU, he was an Assistant Professor at Bennett College from 2006 to 2008, University of Maryland Eastern Shore from 2008 to 2009, and Associate Professor at Beijing Institute of Petro-Chemical Technology, Beijing, China from 1990 to 2000. Dr. Yang has published more than 60 peer-reviewed journal papers and conference proceedings. He has been awarded by NSA, NSF, and DoD with amount of more than half million dollars. His current research interest is computer network and information security.

Lixin Wang is currently a Professor of Computer Science at Columbus State University, Columbus, Georgia. He holds a Ph.D. degree in Computer Science from Illinois Institute of Technology, Chicago IL. His research interests include Network Security, Intrusion Detection Systems, Wireless Networks, Algorithm Design and Analysis. He has been conducting top quality research in these areas and published more than 50 peer-reviewed high-quality research papers, most of which are published in leading journals or top-tier conferences in Computer Science. Since 2011, Dr. Wang has been awarded ten federal grants (from NSA, NSF, US Army Reserve, US Department of Energy, or US Department of Education) as the PI or Co-PI with the total awarded amount more than US$2.6 million. Dr. Wang is currently serving (or served in recent years) as an associate/guest editor for five academic journals in Computer Science, including journal special issues.

Austin Lee (Senior Member, IEEE) is currently an undergraduate student in computer science - cybersecurity track at Columbus State University, Columbus, GA, USA. His research interests include intrusion detection systems and network security.

Peng-Jun Wan received the BS degree from Tsinghua University in 1990, Beijing, China, the MS degree from the Chinese Academy of Sciences, Beijing, China, in 1993, and the PhD degree from the University of Minnesota, Minneapolis, MN, USA, in 1997. He is a fellow of IEEE, and a professor of computer science with the Department of Computer Science, Illinois Institute of Technology, Chicago, IL, USA. His research interests include wireless networks, algorithm design and analysis.