The Loop Fallacy and Serialization in Tracing Intrusion Connections through Stepping Stones

Xinyuan Wang
Cyber Defense Lab
Department of Computer Science
North Carolina State University
Raleigh, NC 27695
USA
xwang5@unity.ncsu.edu

ABSTRACT

Network based intruders seldom attack directly from their own hosts, but rather stage their attacks through intermediate "stepping stones" to conceal their identity and origin. To identify attackers behind stepping stones, it is necessary to be able to trace through the stepping stones and construct the correct intrusion connection chain.

A complete solution to the problem of tracing stepping stones consists of two complementary parts. First, the set of correlated connections that belongs to the same intrusion connection chain has to be identified; second, those correlated connections need to be serialized in order to construct the accurate and complete intrusion connection chain. Existing approaches to the tracing problem of intrusion connections through stepping stones have focused on identifying the set of correlated connections that belong to the same connection chain and have overlooked the serialization of those correlated connections.

In this paper, we use set theoretic approach to analyze the theoretical limits of the correlation-only approach and demonstrate the gap between the perfect correlation-only approach and the perfect solution to the tracing problem of stepping stones. In particular, we identify the serialization problem and the loop fallacy in tracing connections through stepping stones. We formally demonstrate that even with perfect correlation solution, which gives us all and only those connections that belong to the same connection chain, it is still not adequate to serialize the correlated connections in order to construct the complete intrusion path deterministically. We further show that correlated connections, even with loops, could be serialized deterministically without synchronized clock. We present an efficient intrusion path construction method based on adjacent correlated connection pairs.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General – *security and protection (e.g., firewalls)*; K.6.5 [**Management of Computer and Information Systems**]: Security and Protection – *Unauthorized access (e.g., hacking, phreaking)*.

General Terms

Security, Theory.

Keywords

Stepping Stones, Intrusion Tracing, Serialization, Correlation.

SAC'04, March 14–17, 2004, Nicosia, Cyprus.

1. INTRODUCTION

One of most widely used techniques by intruders to hide their origin is to connect through a series of hosts as stepping stones before attacking the final target [7]. For example, an attacker at host A may telnet or ssh into host B, and from there launch an attack against host C. The victim at host C can use IP traceback techniques [3,5 etc.] to find out that the attack comes from host B, but IP traceback can not determine that the attack actually originate from host A behind host B. By laundering through a number of intermediate stepping stones, the attacker makes the source tracing of the attack much more difficult. To identify intruders behind stepping stones, it is critically important to be able to trace the intrusion connections through the stepping stones and construct the correct intrusion connections chain.

A complete solution to the problem of tracing stepping stones includes: 1) the identification of the set of correlated connections that belongs to the same intrusion connection chain; 2) the serialization of the set of correlated connections in order to construct the accurate and complete intrusion connection chain. However, existing approaches to the problem of tracing intrusion connections through stepping stones have focused on identifying the set of correlated connections that belong to the same intrusion connection chain and have left the serialization of correlated connections an afterthought. While finding the right set of correlated connections forms the foundation of solving the tracing problem of intrusion connection chain, it does not, however, completely solve the tracing problem.

In this paper, we use set theoretic approach to analyze the theoretical limits of the correlation-only approach and demonstrate the gap between the perfect correlation solution and the perfect tracing solution of stepping stones problem. In particular, we identify the serialization problem and the loop fallacy in tracing connections through stepping stone. We

formally demonstrate that even with perfect correlation solution, which gives us all and only those connections in the connection chain, it is still not adequate to serialize the complete intrusion connection chain deterministically. This is due to the lack of order information from the set of correlated connections. We show that without deterministic connection serialization, the effectiveness of existing correlation-only approaches for tracing intrusion connections through stepping stones could be seriously affected by one simple practice of the attacker: introducing loops by passing some stepping stone more than once. We further demonstrate that correlated connections, even with loops, could be serialized deterministically without synchronized clock. We present an efficient serialization method based on adjacent correlated connection pairs from each stepping stone.

The remainder of this paper is organized as follows. Section 2 reviews related works on tracing intrusion connections through stepping stones. Section 3 formally formulates the overall problem of tracing intrusion connections through stepping stones and identifies the serialization problem. Section 4 illustrates the loop fallacy in deterministic serialization of correlated connections. Section 5 analyzes the serialization problem and presents the deterministic serialization of correlated connections without synchronized clock. Section 6 concludes this paper.

2. RELATED WORKS

The earliest works (DIDS [4], CIS [2]) on tracing intrusion connections through stepping stones were based on tracking users' login activities at different hosts. Because the attacker who has root control of the stepping stone could easily disguise, delete or forge user login activities at the stepping stone, tracing approaches based on tacking users' login activities at stepping stone could be easily defeated. To overcome this shortcoming, Tracing and correlation approaches based on comparing packet contents [Thumbprinting [6], SWT [10]] have been developed.

To be able to correlate and trace encrypted attack traffic, new generation of network based correlation approaches has been developed, based on the inter-packet timing characteristics (ON/OFF-based [13], Deviation-based [11] and IPD-based [9]). Ideally, the inter-packet timing characteristics of an interactive flow is unique enough and is invariant across routers and stepping stones so that effective correlation could be constructed.

To address the new challenge of active timing perturbation by adversary, Donoho *et al*. [1] have recently studied the theoretical limits of the adverse effects of the active timing perturbation. Wang *et al*. [8] developed a framework for constructing robust timing based correlation scheme against random timing perturbation.

Limitations of Correlation-Only Approach

We have shown that almost all network-based tracing approaches are correlation-only. While the correlation of encrypted attack traffic is till a challenging task due to various active countermeasures used by adversary, there is a limit on the theoretically achievable effectiveness of even the perfect correlation solution.

In the rest of this paper, we investigate the gap between the perfect stepping stone tracing solution and the perfect stepping stone correlation solution, and we show what it takes to fill the gap.

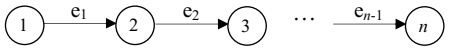

Figure 1: Loopless Linear Connection Chain

3. THE PROBLEM of TRACING INTRUSION CONNECTIONS through STEPPING STONES

In this section, we use set theoretic approach to formulate the overall problem of tracing intrusion connections through stepping stones. We first review the basic concepts of Set Theory we used.

3.1 Ordinals of Basic Set Theory

For binary relation R on set S, we use $Field(R)$ to denote the set of elements of each ordered pair in R. That is $\text{Field}(R)=\{x: \langle x,y\rangle \in R \vee \langle y,x\rangle \in R\}$.

Binary relation R is called

Reflexive:	if $\forall x \in \text{Field}(R)$ [x R x]
Irreflexive:	if $\forall x \in \text{Field}(R)$ [$\neg$(x R x)]
Symmetric:	if $\forall$ x, y$\in$ Field(R) [x R y $\Leftrightarrow$ y R x]
Anti-symmetric:	if $\forall$ x, y$\in$ Field(R) [x R y $\Rightarrow \neg$(y R x)]
Transitive:	if $\forall$ x, y, z$\in$ Field(R) [(x R y $\wedge$ y R z) $\Rightarrow$ x R z]
Linear (connected):	if $\forall$ x, y$\in$ Field(R) [x R y $\vee$ y R x]

Binary relation R on S is a *partial-order* if it is 1) anti-symmetric and 2) transitive. Partial order R on S is a *total-order* if it is linear (connected).

Given partial order R on S and $A \subseteq S$, if there exists $a \in A$ such that $\forall x \in A$ [a R x], we say a is the *R-least* (or *R-minimal*) in A. A total order R on S is a *well-order* on S if every non-empty subset of S has a R-minimal.

3.2 Overall Tracing Problem Model

Given a series of computer hosts H_1, H_2, … H_{n+1} (n>1), when a person (or a program) sequentially connects from H_i into H_{i+1} (i=1,2, … n), we refer to the sequence of connections $\langle c_1, c_2, \ldots c_n\rangle$, where $c_i=\langle H_i, H_{i+1}\rangle$ (i=1, …n), as a *connection chain* on $\langle H_1, H_2, \ldots H_{n+1}\rangle$. Here all c_i's are always distinct, but not all H_i's are always distinct. In case some host appears more than once in sequence $< H_1, H_2, \ldots H_{n+1}>$, there exists loop in the connection chain $\langle c_1, c_2, \ldots c_n\rangle$.

The *tracing problem* of a connection chain (or stepping stone) is, given c_1 of some unknown connection chain $\langle c_1, c_2, \ldots c_n\rangle$ (n>1), to identify $\langle c_1, c_2, \ldots c_n\rangle$.

Any particular connection chain $\langle c_1, c_2, \ldots c_n\rangle$ is an *ordered* set of connections. We refer those connections within same connection chain as *correlated* to each other and corresponding set $\{c_1, c_2, \ldots c_n\}$ as *set of correlated connections* or *correlation set*. This can be formally modeled by a binary relation on the overall connection set. We define binary relation *CORR* on the overall connection set $\hat{C}$ such that

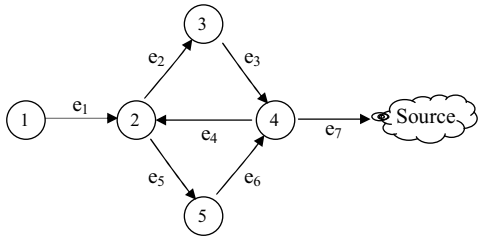

Figure 2: Loop Fallacy in Serializing Correlated Connections

$$\forall c, c' \in \hat{C}\ [c\ CORR\ c'\ iff\ (c \in \{c_1, c_2, \ldots c_n\} \wedge c' \in \{c_1, c_2, \ldots c_n\})] \quad (1)$$

It is obvious that *CORR* is specific to the correlation set and it is 1) self-reflexive; 2) symmetric and 3) transitive. Therefore binary relation *CORR* is an equivalence relation on $\hat{C}$ and it partitions the overall set of connections into a particular set of correlated connections and rest of the connections.

Because connection chain $<c_1, c_2, \ldots c_n>$ is an ordered set, each c_i has an order number $Ord(c_i)$ associated with it. The overall ordering information of $<c_1, c_2, \ldots c_n>$ can be formally modeled by the binary relation $\angle$ on $\{c_1, c_2, \ldots c_n\}$ such that

$$\forall c, c' \in \{c_1, c_2, \ldots c_n\}\ [c \angle c'\ \ iff\ \ Ord(c) < Ord(c')] \quad (2)$$

It is obvious that $\angle$ well orders set $\{c_1, c_2, \ldots c_n\}$ and it uniquely determines $<c_1, c_2, \ldots c_n>$ from $\{c_1, c_2, \ldots c_n\}$.

For any particular connection chain $<c_1, c_2, \ldots c_n>$, there exists unique binary relations *CORR* and $\angle$, which in turn uniquely determines $<c_1, c_2, \ldots c_n>$. Therefore, the overall tracing problem of connection chain can be divided into the following sub-problems:

1) *Correlation Problem*:

   Given c_1 of some unknown connection chain $<c_1, c_2, \ldots c_n>$, identify set $\{c_1, c_2, \ldots c_n\}$; Or equivalently, given any two connections c and c', determine if c *CORR* c'.

2) *Serialization Problem*:

   Given unordered set of correlated connections $C=\{c_1, c_2, \ldots c_n\}$, serialize $\{c_1, c_2, \ldots c_n\}$ into an ordered set $<c_1', c_2', \ldots c_n'>$ ($c_i' \in C$, i=1, … n) such that $c_i' \angle c_{i+1}'$ (i=1, … n-1); Or equivalently, given any two connections c and c', determine if $c \angle c'$ or $c' \angle c$.

Two observations can be made about the overall tracing problem:

1) The result of the serialization problem is based upon the result of the correlation problem

2) The perfect result of the overall tracing problem consists of the perfect result of the correlation problem and the perfect result of the serialization problem based upon the perfect correlation result.

Observation 1) shows the inter-dependency between the correlation problem and the serialization problem, and it explains why existing works on the overall tracing problem have focused on the correlation problem. Observation 2) reveals that while the solution to the correlation problem is the very foundation of the solution to the overall tracing problem, it is not adequate to construct the complete solution to the overall tracing problem. What's missing from the correlation-only approach is the serialization of the correlation result.

In the remainder of this paper, we identify, analyze this gap and we present an efficient solution to the serialization problem.

4. The LOOP FALLACY in DETERMINISTIC SERIALIZATION of CORRELATED CONNECTIONS

Ideally the complete solution of the problem of tracing intrusion connections through stepping stones would give the exact order of the intrusion connections that pass the stepping stones in addition to identifying those correlated connections that belong to the same connections chain. In case there are stepping stones and connections outside the observing area (or scope) of the tracing system, the tracing system should deterministically point out the right direction from which the intrusion comes in. As the stepping stones used by intruders could easily be thousands miles apart and under different jurisdiction, it is critically important to be able accurately point out the right direction from which the intrusion comes from outside the current tracing system to make the tracing system useful in real-world.

Unfortunately, even with perfect correlation solution, which gives all and only those correlated connections within the observing scope that belong to the same connection chain, it is still not adequate to deterministically construct the complete intrusion path or even find the right direction from which intrusion comes in.

In case the intrusion connection passes each stepping stone only once each stepping stone has only one incoming and outgoing connection, and there is only one way to serialize those correlated connections to construct the intrusion path as shown in Figure 1.

However, when some stepping stones are passed more than once, there exists loop or cycle in the intrusion connection chain, and there are more than one ways to serialize those correlated connections. Figure 2 shows an example of intrusion connection chain with multiple stepping stones, where node 1 is the intrusion target and e_1, e_2, e_3, e_4, e_5, e_6, e_7 are the backward connections from the intrusion target toward the source of the intrusion. A perfect correlation solution would report that e_1, e_2, e_3, e_4, e_5, e_6, e_7 are correlated and belong to the same intrusion connection chain. Given the knowledge that node 1 is the intrusion target, we know that the intrusion to node 1 comes from node 2 as there is only one correlated connection e_1 between node 1 and node 2. However, node 2 has two outgoing connections e_2 and e_5 that are part of same connection chain, and there are multiple ways to serialize those correlated connections. Furthermore, when some stepping stones are outside of the observing area of the tracing system, loops in the intrusion connection chain could introduce dilemma in determine the right direction from which the intrusion comes in. Figure 3 shows two such examples. When node 3,4,5 are outside the observing area of the tracing system, node 2 sees two correlated outgoing connections e_2 and e_5. Without additional information, there is no way for node 2 to determine which connection points to the host that is closer to the intrusion source. When node 3 is out of the observing scope, there are multiple ways to serializing the correlated connections, which point to

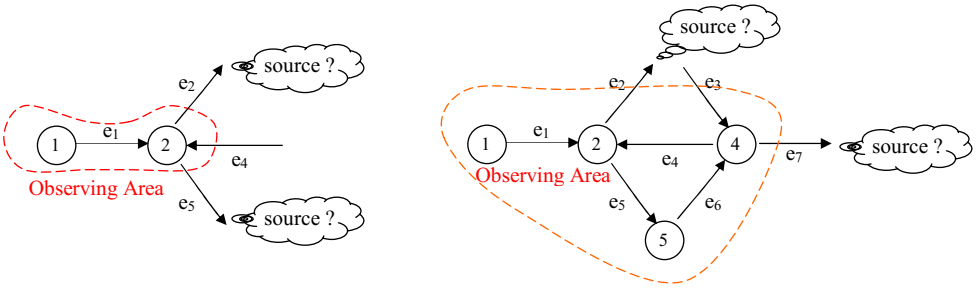

Figure 3: Tracing Dilemma with Limited Observing Area

different directions to the intrusion source. For example, both serialization $<e_1, e_2, \ldots e_3, e_4, e_5, e_6, e_7>$ and $<e_1, e_5, e_6, e_7, \ldots e_3, e_4, e_2>$ are possible, which imply e_7 and e_2 respectively as the connections pointing to the intrusion source.

These examples indicate that correlation only approach is a partial solution to the problem of tracing intrusion connections through stepping stones. What is missing from the correlation only solution is the serialization of those correlated connections. It is this phenomenon – that people in general do not take the potential loops or cycles of intrusion connection chain into account when intuitively solving the tracing problem with correlation only approaches – that is named "the loop fallacy" in tracing intrusion connections through stepping stones.

5. DETERMINISTIC SERIALIZATION of CORRELATED CONNECTIONS

We have shown in previous section, the set of correlated connections itself is not adequate to serialize those correlated connections deterministically. In order to deterministically serialize correlated connections, some additional information on the correlated connection is needed.

One possible way to serialize correlated connection is use globally synchronized time-stamp to determine the relative order of correlated connections. However, collecting precisely synchronized timestamp on all connections across the internet is difficult due to the following reasons: 1) not all the hosts on the internet have precise clock synchronization; 2) dynamic network delay (which may cause out-of-order delivery) complicates distributed timestamping; 3) distributed clock synchronization is also subject to malicious attacks.

Another way to serialize correlated connections is based on adjacency or causal relationship. Compared with timestamp based approach, adjacency based approach does not require any global clock synchronization at all and is robust against network delay jitters.

In this section, we focus on solving the problem of deterministic serialization of correlated connections without global clock synchronization. We use set theoretic approach to formally establish that while the set of correlated connection itself is not adequate to serialize those correlated connections, the set of adjacent correlated connection pairs of each stepping stone is sufficient to serialize those correlated connection deterministically even if there is loops with the connection chain.

Given a set of correlated connections C, it can be thought as a set of edges of a directed graph DG such that DG=<V, E>, V={x: $\exists <x, y> \in C \vee \exists <y, x> \in C$} and E=C. We assume that there is no *self-loop edge* in DG, that is $\forall <u, v> \in E\ [u \neq v]$. Therefore, the serialization of elements of C can be represented by the ordering of elements of either V or E.

We use $u \rightarrow v$ to represent that there is directed path from u to v. and we define DG to be *one-way connected* if: $\forall u, v \in V\ [\exists\ u \rightarrow v \vee \exists v \rightarrow u]$, and DG to be *edge one-way connected* if: $\forall <u_1, v_1>, <u_2, v_2> \in E\ [v_1 \rightarrow u_2 \vee v_2 \rightarrow u_1]$.

One necessary condition for the serialization of correlated connections to be correct is that the ordering of the correlated connections maintains the one-way connectivity of the edges and end-points of correlated connections.

5.1 Point Connectivity and Serialization Based on Point Adjacency

We first consider serialization of correlated connections based on point adjacency property of those correlated connections.

We define **Point-Adjacency** (**P-Adj**) on V as the binary relation $\{<u, v>: <u, v> \in E\}$. It is easy to see that P-Adj is irreflexive and it models the adjacency relation among the elements of V.

We define **Point Connectivity** (**PC**) as the binary relation on V, such that

1) $<u, v> \in E\ [u\ PC\ v]$

2) $u, v, w \in V\ [(u\ PC\ v \wedge v\ PC\ w) \Rightarrow u\ PC\ w]$

Therefore binary relation PC is the *transitive-closure* of P-Adj. Because DG has no self-loop edge, PC is anti-symmetric, thus PC is a partial order on V. Here we use $<_{PC}$ to represent PC. If there exists some $v \in V$, such that $\forall\ u \in V\ [u \neq v \Rightarrow v <_{PC} u]$, we define such an element v as **PC-minimal** on V.

From the definitions, it is easy to see that given a DG, there is only one P-Adj and $<_{PC}$ defined on V.

Here binary relation $<_{PC}$ formally models the directed connectivity among the vertices in V and $u <_{PC} v$ *iff* there exists a path from u to v.

THEOREM 1: the necessary and sufficient conditions for $<_{PC}$ to be well-order on V are:

1) DG=<V, E> is one-way connected

2) DG has no directed cycles

PROOF:

Sufficiency:

Given that DG has no directed cycles, $<_{PC}$ is anti-symmetric: $\forall u, v \in V\ [u <_{PC} v \Rightarrow \neg (v <_{PC} u)]$. Because DG is one-way connected, $<_{PC}$ is transitive. Therefore $<_{PC}$ is a partial order on V.

Given DG is one-way connected, $\forall u, v \in V\ (u \neq v)$, there exist a directed path either $u \rightarrow v$ or $v \rightarrow u$. We have either $u <_{PC} v$ or $v <_{PC} u$. Therefore, $<_{PC}$ is a total-order on V.

Assume $<_{PC}$ is not a well-order on V, then there exist a non-empty set of vertices $V' \subseteq V$ such that V' does not have PC-minimal. That is $\forall v \in V'$, $\exists u \in V'$ such that $u <_{PC} v$. We list elements of V', starting from $\forall v_1 \in V'$, and adding $v_{i+1} \in V'$ to the left of $v_i \in V'$ if $v_{i+1} <_{PC} v_i$ and $v_{i+1} \notin \{v_i, v_{i-1} \ldots v_1\}$ as following:

$$v_n \ldots v_{i+1}\ v_i \ldots v_2\ v_1$$

Because V' is finite, the above list is also finite. Assume the left-most element of above list is v_n, we have v_i $(1 \leq i < n)$ such that $v_i <_{PC} v_n$, therefore $<v_i, v_n, \ldots v_i>$ forms a directed cycle in G. This contradicts condition 2). Therefore $<_{PC}$ well-orders V.

Necessity:

1) Because $<_{PC}$ is well-order on V, it is total-order on V. $\forall u, v \in V\ (u \neq v)$, we have either $u <_{PC} v$ or $v <_{PC} u$. Then there exist a path either $u \rightarrow v$ or $v \rightarrow u$. Therefore DG is one-way connected.

2) Assume DG has directed cycle of $n>1$ vertices: $v_n \ldots v_2\ v_1$, consider non-empty subset of V $\{ v_n \ldots v_2\ v_1\}$, there is no PC-minimal in that set. This contradicts with the prerequisite that $<_{PC}$ well-orders V. Therefore DG has no directed cycle.

Because an intrusion connection chain may pass a particular stepping stone more than once, which introduces directed cycles in the connection chain, the serialization of end points of correlated connections based on point adjacency is not deterministic.

5.2 Edge Connectivity and Serialization Based on Edge Adjacency

We now consider serialization of correlated connections based on edge adjacency relation among those correlated connections.

We define **Edge-Adjacency** (**E-Adj**) on E as the binary relation: $\{<<u, v>, <v, w>> : <u, v>, <v, w> \in E\}$. It is easy to see that E-Adj is irreflexive and it models the adjacency relation among the elements of E.

We define **Edge Connectivity** (**EC**) as the binary relation on E, such that

1) $<u, v>, <v, w> \in E\ [<u, v>\ EC\ <v, w>]$

2) $<u_1, v_1>, <u_2, v_2>, <u_3, v_3> \in E\ [(<u_1, v_1>\ EC\ <u_2, v_2> \wedge <u_2, v_2>\ EC\ <u_3, v_3>) \Rightarrow <u_1, v_1>\ EC\ <u_3, v_3>]$

Therefore binary relation EC is the *transitive-closure* of E-Adj. Because each correlated connection is distinct, EC is anti-symmetric, thus EC is a partial order on E. Here we use $<_{EC}$ to represent EC. IF there exists some $<u_1, v_1> \in E$, such that $\forall <u_2, v_2> \in E\ [<u_1, v_1> \neq <u_2, v_2> \Rightarrow <u_1, v_1> <_{EC} <u_2, v_2>]$, we define $<u_1, v_1>$ as **EC-minimal** on E

From the definitions, it is easy to see that given a DG, there is only one E-Adj and $<_{EC}$ defined on E.

Binary relation $<_{EC}$ also models the directed connectivity among vertices of V and $<u_1, v_1> <_{EC} <u_2, v_2>$ *iff* there exists a path from v_1 to u_2.

THEOREM 2: the necessary and sufficient conditions for $<_{EC}$ to be well-order on E are:

1) DG=<V, E> is one-way connected

2) DG has no directed cycles

3) DG has no out-branch: $\forall v \in V$ (v has at most single successor)

PROOF:

Sufficiency:

Given $\forall <u_1, v_1>, <u_2, v_2> \in E$ and $<u_1, v_1> \neq <u_2, v_2>$, we have $u_1 \neq u_2$ because of 3).

Assume $v_1 = v_2$. Consider $u_1, u_2 \in V$, because of 1), there exists path: $u_1 \rightarrow u_2$. Because of 3) we have $v_1 \rightarrow u_2$, that is $v_2 \rightarrow u_2$. Then we have a cycle $<v_2, u_2, v_2>$, and it contradicts condition 2). Therefore $v_1 \neq v_2$.

Assume $v_1 \rightarrow u_2$, because of condition 2), there is no path from u_2 to v_1 (otherwise we have a loop). That is $\forall <u_1, v_1>, <u_2, v_2> \in E\ [<u_1, v_1> <_{EC} <u_2, v_2> \Rightarrow \neg (<u_2, v_2> <_{EC} <u_1, v_1>)]$. Therefore, $<_{EC}$ is a partial order on E.

Assume there is neither path from v_1 to u_2 nor path from v_2 to u_1. Because of 1), we have $u_2 \rightarrow v_1$ and $u_1 \rightarrow v_2$. Because of 3), we have $v_2 \rightarrow v_1$ and $v_1 \rightarrow v_2$. That forms a cycle, which contradicts condition 2). Therefore there is either $v_1 \rightarrow u_2$ or $v_2 \rightarrow u_1$. That is equivalent to either $<u_1, v_1> <_{EC} <u_2, v_2>$ or $<u_2, v_2> <_{EC} <u_1, v_1>$. Therefore $<_{EC}$ is a total-order on E.

Assume $<_{EC}$ is not well-order on E, then there exist a non-empty set $E' \subseteq E$ such that there is no EC-minimal on E'. That is $\forall <u_1, v_1> \in E'$, $\exists <u_2, v_2> \in E'$ such that $<u_2, v_2> <_{EC} <u_1, v_1>$. We list elements E', starting from $\forall <u_1, v_1> \in E'$, and adding $<u_{i+1}, v_{i+1}> \in E'$ to the left of $<u_i, v_i> \in E'$ if $<u_{i+1}, v_{i+1}> <_{EC} <u_i, v_i>$ and $<u_{i+1}, v_{i+1}> \notin \{<u_i, v_i>, <u_{i-1}, v_{i-1}> \ldots <u_1, v_1>\}$ as following:

$$<u_n, v_n> \ldots <u_{i+1}, v_{i+1}> <u_i, v_i> \ldots <u_2, v_2> <u_1, v_1>$$

Because E' is finite, the above list is also finite. Assume the left-most element of above list is $<u_n, v_n>$, we have $<u_i, v_i>$ $(1 \leq i < n)$ such that $<u_i, v_i> <_{EC} <u_n, v_n>$, therefore $<<u_i, v_i>, <u_n, v_n>, \ldots <u_i, v_i>>$ forms a directed cycle in DG. This contradicts condition 2). Therefore $<_{EC}$ well-orders E.

Necessity:

1) $\forall u, v \in V\ (u \neq v)$, there exist $e_1, e_2 \in E$ $(e_1 \neq e_2)$ such that u is endpoint of e_1 and v is endpoint of e_2. Without losing generality, we assume $e_1 = <u, x>$ and $e_2 = <v, y>$. Because EC well-orders E, it total-orders E. Therefore $e_1 <_{EC} e_2$ or $e_2 <_{EC} e_1$. There exists path either from u to v or from v to

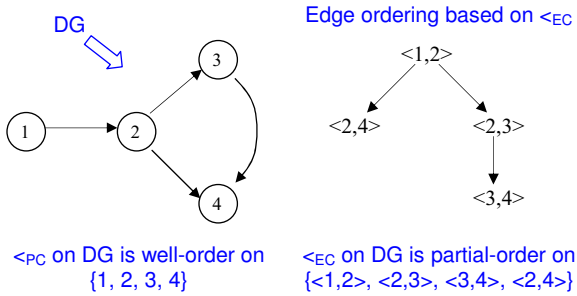

Figure 4: Point Connectivity $<_{PC}$ and Edge Connectivity $<_{EC}$

u in G.

2) Assume G has directed cycle of n >1 edges: $<v_1, v_2>$, $<v_2, v_3>$, … $<v_n, v_1>$, consider non-empty subset of E $\{<v_1, v_2>, <v_2, v_3>$, … $<v_n, v_1>\}$, there is no PDEC-minimal in that set. This contradicts with the prerequisite that $<_{EC}$ well-orders E. Therefore DG has no directed cycles.

3) Assume DG has out-branch: $\exists <u, x>, <u, y> \in E$ $(x \neq y)$. Because $<_{EC}$ well-orders E, we have either $<u, x> <_{EC} <u, y>$ or $<u, y> <_{EC} <u, x>$. Without losing generality, we assume $<u, x> <_{EC} <u, y>$, then there exist path $x \rightarrow u$. $\{x, \ldots u, x\}$ forms a cycle, which contradicts the necessary condition 2) just proved. Therefore DG has no out-branch: $\forall v \in V$ (v has single successor).

Please be noted that given DG=<V,E>, in order for $<_{EC}$ to well-orders E, DG must have no out-branch, which is not required for to $<_{PC}$ well-order V. Figure 4 shows such an example, where $<_{PC}$ well-orders {1,2,3,4} and $<_{EC}$ is not even a total-order on E as <2,3> and <2,4> have no relative order.

Because no directed cycles is a necessary condition for $<_{EC}$ to be well-order on E, the serialization of correlated connections based on edge adjacency is not deterministic either. Figure 5 shows an example of serialization of connections based on edge adjacency.

5.3 Serialization Based on Adjacent Connection Pairs

We have demonstrated that the ordering of correlated connections based on adjacency is not always deterministic and unique. When the intrusion connection chain has loops or cycles, there are multiple ways to serialize those correlated connections while keeping the connectivity. This dilemma is due to the fact that there could be more than two connections adjacent to each other through one vertex and the set of correlated connections gives no clue about how to pair match those adjacent connections (shown in Figure 6).

To serialize the correlated connections deterministically, we need information about how the adjacent connection are pair matched. This is modeled by the concept of adjacent connection pair.

Given a connection chain $<H_1, H_2, \ldots H_n>$, we define $<<H_i, H_{i+1}>, <H_{i+1}, H_{i+2}>>$ (i=1,2, … n-2) as the *adjacent connection pair* on H_{i+1}. Please note that there could be $H_i=H_j$ ($1 \le i,j \le n$, $i \neq j$). Adjacent connection pair carries the order information about the two adjacent connections on a particular vertex. We use **PE-Adj** to represent the set of adjacent connection pairs.

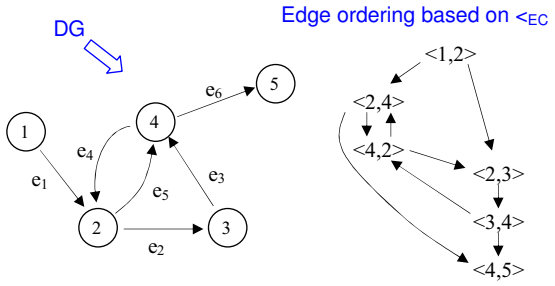

Figure 5: Edge Serialization Based on Edge Connectivity $<_{EC}$

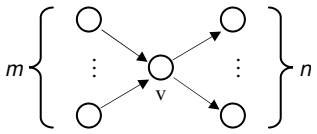

Figure 6: *m* Incoming Connections Adjacent to *n* Outgoing Connections

Given a set of adjacent connection pairs PE-Adj, we can construct the set of connection

$$E_{PE\text{-}Adj}=\{e: \exists <e, e_i> \in \text{PE-Adj} \vee \exists <e_j, e> \in \text{PE-Adj}\}$$

and the set of vertices

$$V_{PE\text{-}Adj}=\{v: \exists <<u, v>, <v, w>> \in \text{PE-Adj} \vee \exists <<v, u>, <u, w>> \in \text{PE-Adj} \vee \exists <<u, w>, <w, v>> \in \text{PE-Adj}\}$$

and the directed graph $DG=<V_{PE\text{-}Adj}, E_{PE\text{-}Adj}>$. Therefore PE-Adj is binary relation on $E_{PE\text{-}Adj}$ and PE-Adj $\subseteq$ E-Adj on $E_{PE\text{-}Adj}$.

We define binary relation **Paired Edge Connectivity** (**PEC**) on $E_{PE\text{-}Adj}$, such that

1) $<<u, v>, <v, w>> \in$ PE-Adj $[<u, v>$ PEC $<v, w>]$

2) $<u_1, v_1>, <u_2, v_2>, <u_3, v_3> \in E_{PE\text{-}Adj}$ [$(<u_1, v_1>$ PEC $<u_2, v_2> \wedge <u_2, v_2>$ PEC $<u_3, v_3>) \Rightarrow <u_1, v_1>$ PEC $<u_3, v_3>$]

Because each correlated connection is distinct, PEC is anti-symmetric, thus it is a partial order on E. Here we use $<_{PEC}$ to represent PEC. If there exists $<u_1, v_1> \in$ E, such that $\forall <u_2, v_2> \in$ E $[<u_1, v_1> \neq <u_2, v_2> \Rightarrow <u_1, v_1> <_{PEC} <u_2, v_2>$], we define $<u_1, v_1>$ as **PEC-minimal** on E

Element of PE-Adj, $<e_i, e_j>$, can also be thought as a directed edge whose endpoints (tail and head) are e_i and e_j, from which another directed graph can be deterministically constructed.

We define the *paired line graph* of DG, written as PL(DG), as the directed graph whose vertices are the edges of DG, with $<e_i, e_j> \in$ E(PL(DG) when $<e_i, e_j> \in$ PE-Adj.

It is obvious that V(PL(DG)) $\equiv$ E(DG) $\equiv E_{PE\text{-}Adj}$, therefore PE-Adj on DG corresponds to P-Adj on PL(DG) and $<_{PEC}$ on DG

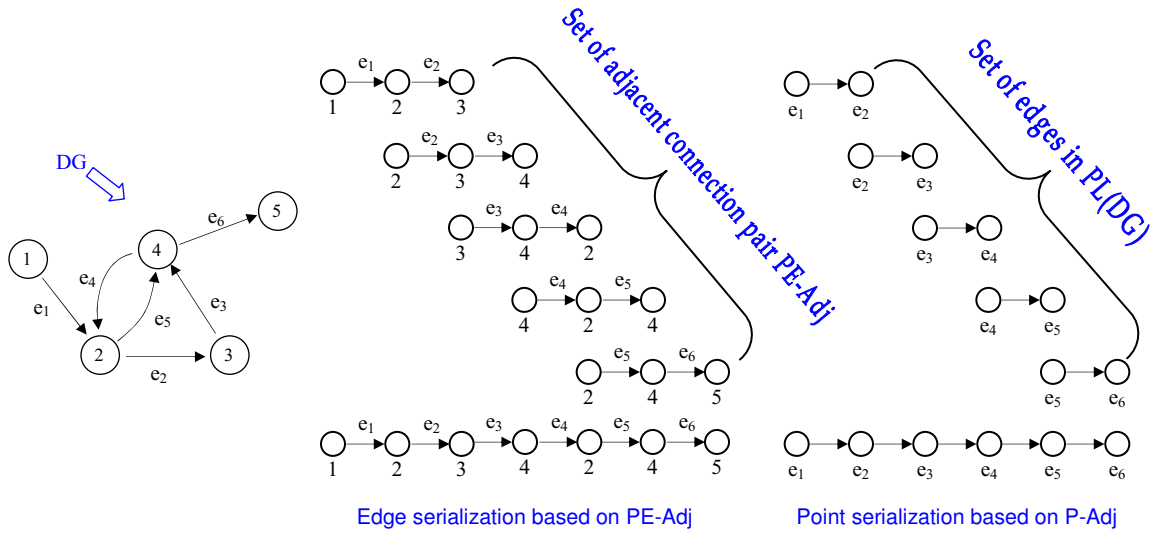

Figure 7: Edge Serialization Based on Adjacent Connection Pair PE-Adj

corresponds to $<_{PC}$ on PL(DG).

We further define *reachable set* of a particular edge $e \in E_{PE\text{-}Adj}$ as $RS_{PE\text{-}Adj}(e)=\{e_i: e <_{PEC} e_i\}$. PE-Adj is *edge one-way connected iff* $\forall e_i, e_j \in E_{PE\text{-}Adj} (e_i <_{PEC} e_i \vee e_i <_{PEC} e_i)$. PE-Adj is *loop less* iff $\forall e \in E_{PE\text{-}Adj} [e \notin RS_{PE\text{-}Adj}(e)]$.

We say PE-Adj is loop less if any connection within the set of adjacent connection pair will not reach itself through the adjacent connection pair.

THEOREM 3: If EP-Adj is edge one-way connected and loop less, $<_{PEC}$ well-orders $E_{PE\text{-}Adj}$.

PROOF:

Because PE-Adj is edge one-way connected, $<_{PEC}$ total-orders $E_{PE\text{-}Adj}$.

Assume DG=< $V_{PE\text{-}Adj}$, $E_{PE\text{-}Adj}$>, consider the paired line graph of DG: PL(DG)=<V, E>, where V=$E_{PE\text{-}Adj}$ and E=PE-Adj. $<_{PEC}$ total-orders $E_{PE\text{-}Adj}$ corresponds to $<_{PC}$ on V total-orders V. Therefore PL(DG) is one-way connected.

Because PE-Adj is loop less, $\forall v \in V$, it won't reach v again in PL(DG). That is PL(DG) has no directed cycle.

Apply theorem 1, $<_{PC}$ well-orders V on PL(DG), which corresponds to $<_{PEC}$ well-orders $E_{PE\text{-}Adj}$.

If PE-Adj contains all the adjacent connection pair from every stepping stone along the connection chain, PE-Adj is edge one-way connected. PE-Adj is also loop less because each connection within the connection chain is unique while it correlates with others.

Therefore, the complete and accurate intrusion connection chain can be constructed deterministically from the set of adjacent correlated connection pairs, even if there are loops within the connection chain. Figure 7 illustrate an example of the deterministic serialization of correlated connections from the set of adjacent correlated connection pairs.

5.4 Finding Adjacent Correlated Connection Pairs

We say that the set of correlated connection pairs is with regard to (*wrt*) connection c if c is correlated with all connections that form the correlated connection pairs.

The set of correlated connection pairs (with regard to connection c) can be constructed by union of each subset collected at each stepping stone.

The subset of correlated connections pairs at each stepping stone can be constructed at real-time by the following algorithm:

1) For each new incoming (or outgoing) connection I_i (or O_i) that is not self-loop, record I_i (or O_i) into queue Q: x_1, x_2, …x_{i-1}, where x_j ($1 \le j \le i$-1)could be either incoming or outgoing connection.

2) Using correlation approach to find those, if any, connections that are correlated with c, from all the connections recorded in Q.

3) Extract those correlated connections, in sequence, from Q into correlation queue Q_c.

4) Assume Q_c has c_1, c_2, …c_m, if c_1 is incoming connection, the subset of correlated connection pairs is $\{< c_1, c_2 >, < c_3, c_4 >, \ldots < c_{2\times\lfloor m/2 \rfloor-1}, c_{2\times\lfloor m/2 \rfloor} >\}$; if c_1 is outgoing connection, the subset of correlated connection pairs is $\{< c_2, c_3 >, < c_4, c_5 >, \ldots < c_{2\times\lfloor (m-1)/2 \rfloor}, c_{2\times\lfloor (m-1)/2 \rfloor+1} >\}$.

The correctness of the algorithm is guaranteed by the following property of $Q_c = c_1, c_2, \ldots c_m$: if c_i is incoming connection, then c_{i+1} is outgoing connection; if c_i is outgoing connection, then c_{i+1} is incoming connection.

Therefore, in order to construct the set of correlated connection pairs, we just need to record the start of all the incoming and outgoing correlated connection at each stepping stone in sequence, from which we can construct the subset of correlated

connection pairs of that stepping stone. Then we can construct the whole set of correlated connection pairs by summation of all the subsets regarding to the same correlation.

For example, assume the sequence of the backward traffic from the attack target to the attack source showed in Figure 3 is $<e_1, e_2, e_3, e_4, e_5, e_6, e_7>$. By the applying the first three steps of the algorithm described above, node 2 will have its $Q_c = e_1, e_2, e_4, e_5$, and node 4 will have its $Q_c = e_3, e_4, e_6, e_7$. After step 4, node 2 will have set of correlated connection pairs: $\{<e_1, e_2>, <e_4, e_5>\}$, and node 4 will have set correlated connection pairs: $\{<e_3, e_4>, <e_6, e_7>\}$. While node 2 does not know how many stepping stones might exist between e_2 and e_4, it knows e_5 is the connection that is closest to the attack source from its point of view. Similarly, node 4 knows e_7 is the connection that is closest to the attack source from its point of view. In case node 1,2,3,4,5 are all within the tracing system, a complete and accurate intrusion path over node 1,2,3,4,5 can be constructed deterministically.

6. CONCLUSIONS

Tracing network based intruders behind stepping stones is a challenging problem, especially when the intrusion connection chain passes some stepping stone multiple times in attempt to further disguise its intrusion path and source.

In this paper, we used set theoretic approach to investigate the gap between the perfect stepping stone correlation solution and perfect stepping stone tracing solution. We first identified the largely overlooked serialization problem and the loop fallacy in tracing intrusion connections though stepping stones. Existing approaches to the tracing problem of stepping stones have focused on correlation only and have left the serialization of correlated connections as an afterthought. We demonstrated that even the perfect correlation solution, which gives all and only those correlated connections, is not sufficient to construct the complete intrusion path deterministically, when there is loop in the intrusion connection chain. We further showed that the complete intrusion path can be constructed deterministically from the set of correlated connection pairs, no matter whether there is any loop in the connection chain or not. We presented an efficient algorithm to construct the set of correlated connection pairs and effective method to serialize correlated connections without global clock synchronization.

The solution of serialization is based upon the correlation result, and the correlation of connections through stepping stones is still a challenging and ongoing research task. Our serialization solution helps to increase the effectiveness of existing correlation result. We view our results as complementary to existing correlation approaches in solving the overall problem of tracing intrusion connections through stepping stones.

7. ACKNOWLEDGMENTS

The author of this paper would like to thank the anonymous reviewers for their helpful comments.

8. REFERENCES

[1] D. Donoho, A.G. Flesia, U. Shanka, V. Paxson, J. Coit and S. Staniford. Multiscale Stepping Stone Detection: Detecting Pairs of Jittered Interactive Streams by Exploiting Maximum Tolerable Delay. In *Proceedings of the 5th International Symposium on Recent Advances in Intrusion Detection (RAID'2002) LNCS-2516*, Pages 17–35, October, 2002.

[2] H. Jung, et al. Caller Identification System in the Internet Environment. In *Proceedings of 4th USENIX Security Symposium*, 1993.

[3] S. Savage, D. Wetherall, A. Karlin and T. Anderson. Practical Network Support for IP Traceback. In *Proceedings of the ACM SIGCOMM '2000*, Pages 295–306, September 2000.

[4] S. Snapp, et all. DIDS (Distributed Intrusion Detection System) – Motivation, Architecture and Early Prototype. In *Proceedings of the 14th National Computer Security Conference*, Pages 167–176, 1991.

[5] D. Song and A. Perrig. Advanced and Authenticated Marking Scheme for IP Traceback. In *Proceedings of IEEE INFOCOM'2001*, Pages 878–886, April 2001.

[6] S. Staniford-Chen, L. T. Heberlein. Holding Intruders Accountable on the Internet. In *Proceedings of IEEE Symposium on Security and Privacy*, Pages 39–49, 1995.

[7] C. Stoll. The Cuckoo's Egg: Tracking Spy through the Maze of Computer Espionage. Pocket Books, October 2000.

[8] X. Wang, D. S. Reeves. Robust Correlation of Encrypted Attack Traffic through Stepping Stones by Manipulation of Interpacket Delays. In *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS 2003)*, Pages 20–29, October 2003.

[9] X. Wang, D. S. Reeves and S.F. Wu. Inter-Packet Delay-Based Correlation for Tracing Encrypted Connections through Stepping Stones. In D. Gollmann, G. Karjoth and M. Waidner, editors, *7th European Symposium on Research in Computer Security (ESORICS'2002) LNCS-2502*, Pages 244–263, October 2002.

[10] X. Wang, D. S. Reeves, S. F. Wu and J. Yuill. Sleepy Watermark Tracing: An Active Network-Based Intrusion Response Framework. In *Proceedings of 16th International Conference on Information Security (IFIP/Sec'01)*, Pages 369–384, June, 2001.

[11] K. Yoda and H. Etoh. Finding a Connection Chain for Tracing Intruders. In F. Guppens, Y. Deswarte, D. Gollmann and M. Waidner, editors, *6th European Symposium on Research in Computer Security (ESORICS'2000) LNCS-1895*, Pages 191–205, October 2000.

[12] K.H. Yung. Detecting Long Connection Chains of Interactive Terminal Sessions. In *Proceedings of the 5th International Symposium on Recent Advances in Intrusion Detection (RAID'2002) LNCS-2516*, Pages 1–16, October, 2002.

[13] Y. Zhang and V. Paxson. Detecting Stepping Stones. In *Proceedings of the 9th USENIX Security Symposium*, Pages 171–184, 2000.