PACKET SCHEDULING AGAINST STEPPING-STONE ATTACKS WITH CHAFF

Ting He, Parvathinathan Venkitasubramaniam, and Lang Tong[†]

School of Electrical and Computer Engineering
Cornell University
Email: {th255, pv45, lt35}@cornell.edu

ABSTRACT

We consider scheduling packet transmissions in a network so that the efficiency of stepping-stone attacks can be severely restrained with the help of stepping-stone monitors. We allow the attacker to encrypt and pad the packets, perturb the timing of packets, and insert chaff packets, but the timing perturbation is subject to a maximum delay constraint. We show that if we randomize packet transmissions, then the attacker has to insert a large amount of chaff to completely evade detection. In particular, if all transmissions are scheduled according to Poisson processes, then we show that the fraction of attacking packets in the attacker's traffic decreases exponentially with the length of the intrusion path.

Index Terms - Stepping-stone attack, Network defense, Scheduling.

1. INTRODUCTION

Stepping-stone attacks are indirect network attacks in which attacking commands are relayed through compromised hosts called "stepping stones" [1]. Since each stepping stone host only sees its immediate predecessor and the victim only sees the last host, it is very difficult to find the true origin of such attacks. The key to defending against stepping-stone attacks is to find the intrusion path.

Although numerous detection schemes have been developed to detect stepping-stone connections, a sophisticated attacker can modify his traffic to evade detection. In particular, he can encrypt and pad the packets so that no information is revealed by the bit patterns or the lengths of packets; he can also perturb the timing of packets by adding random delay or packet reshuffling. Furthermore, the attacker can repacketize the commands, or mix attacking traffic with other traffic or dummy traffic called "chaff". The insertion of chaff makes the detection of stepping-stone traffic especially challenging. We refer to the traffic of attacking packets as *attacking traffic*, and the mixture of attacking traffic and chaff *stepping-stone traffic*.

1.1. Related Work

Staniford and Heberlein [1] are the first to consider the problem of detecting stepping-stone connections. Early techniques are based on the content of the traffic. See, *e.g.*, [1,2]. These techniques, however, are not applicable to detecting encrypted connections. An alternative is to exploit timing characteristics of the traffic; examples include [3–5]. The drawback of these schemes is that they are vulnerable to active timing perturbation by the attacker.

There are a few results on detecting encrypted, timing perturbed stepping-stone connections; see [6–9]. The key assumption of these methods is that the attacker is able to perform a packet-conserving transformation on his traffic, but the transformation is subject to certain constraints.

Packet conservation is too limited to be satisfied in practice. A more general category of stepping-stone connections is the one allowing the attacker to mix attacking traffic with non-attacking traffic, including dummy traffic called chaff. We are only aware of a few results dealing with the attacker's chaff evasion. Peng *et al.* in [10] propose an active detection scheme which combines watermarking with packet matching to detect stepping-stone traffic in chaff. They assume packets have bounded delays, and chaff only appears in the downstream flow. Their scheme injects watermarks in the upstream flow, and finds a subsequence in the downstream flow, whose watermark is closest to the injected one. Such a scheme, however, requires the active manipulation of traffic. Donoho *et al.* [6] point out that in principle it is possible to correlate stepping-stone traffic even if both (bounded) delay and independent chaff are introduced during the relay. Blum *et al.* [8] propose an algorithm called "DETECT-ATTACKS-CHAFF" (DAC) to detect stepping-stone traffic with limited chaff when attacking traffic has bounded delay and bounded peak rate. Algorithm DAC monitors the difference in the number of packets in the incoming and the outgoing streams, and makes detection if the difference exceeds a certain threshold. Algorithm DAC achieves robustness against a limited number of chaff packets by choosing a threshold larger than neces-

[†]This work is supported in part by TRUST (The Team for Research in Ubiquitous Secure Technology), which receives support from the National Science Foundation (NSF award number CCF-0424422) and the following organizations: Cisco, ESCHER, HP, IBM, Intel, Microsoft, ORNL, Qualcomm, Pirelli, Sun and Symantec, and the U. S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2-0011. The U. S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon.

sary. The drawback is that the increase of threshold causes increased false alarm probability, and the attacker can still evade detection by adding a fixed number of chaff packets. In a recent paper [11], Zhang *et al.* propose packet matching schemes to detect stepping-stone traffic with bounded delay perturbation and chaff. They propose to match every arrival with the first departure subject to causality and the delay constraint. They prove that this strategy has exponentially decaying false alarm probability for independent Poisson streams. Their schemes can detect stepping-stone traffic if chaff is only inserted in the departing stream. If chaff can be inserted in the incoming stream, however, one chaff packet suffices to evade their schemes.

1.2. Summary of Results and Organization

In this paper, we show that there are fundamental limits to stepping-stone attacks even if the attacker can encrypt and pad the packets, perturb the timing, and mix attacking packets with chaff. Based on these limits, we propose a randomized packet scheduling strategy to make the defense against stepping-stone attacks more efficient.

We consider encrypted stepping-stone attacks with bounded delay perturbation and chaff. We first analyze the fundamental limits on how fast the attacker can send attacking traffic without being detected by any stepping-stone detector. We propose optimal strategies to schedule the transmission of attacking packets for given realizations of arrival processes while inserting the minimum number of chaff packets. Then the fundamental limits on the rate of the attacking traffic are obtained by characterizing the performance of the proposed chaff-inserting algorithms. We show that although the attacker does not lose much rate in one-hop stepping-stone attacks, the rate of attacking traffic decreases exponentially as the number of hops increases. This result suggests that in detecting stepping-stone traffic, we should jointly consider streams at multiple locations rather than doing local detection separately.

We then compare the achievable rates of the attacking traffic under randomized packet scheduling versus deterministic scheduling. The comparison suggests that randomized packet transmissions can make the network much more robust to stepping-stone attacks.

The rest of the paper is organized as follows. Section 2 defines the problem. Section 3 gives a limit on the rate of attacking traffic passing through a single stepping-stone host. In Section 4, the result is generalized to the case of multiple stepping-stone hosts. Section 5 presents how randomized packet scheduling can facilitate stepping-stone detection. Finally, Section 6 concludes the paper with comments on its limitation.

2. PROBLEM STATEMENT

Let the packet arrivals on stream i be represented by a point process

$$S_i = (s_1^{(i)},\ s_2^{(i)},\ s_3^{(i)}, \ldots), \quad i = 1,\ 2, \ldots$$

where $s_k^{(i)}$ is the kth arrival epoch of stream i. Let $\mathcal{T}_i = \{s_1^{(i)},\ s_2^{(i)}, \ldots\}$ be the set of the elements in S_i. Let S_1 be an incoming stream of the first host, and S_{i+1} ($i = 1, \ldots,\ n$) be a outgoing stream at the ith host. Normally, the outgoing stream at the ith host is different from the incoming stream at the $i+1$th host due to perturbations from clock skews and propagation delay, but we assume that these perturbations are known so that the streams can be adjusted to make them roughly the same. The adjustment also makes sure that streams collected at different hosts are comparable. Such adjustment can be done by sending training packets.

Normally, S_i's are independent. If, however, $(S_i)_{i=1}^{n+1}$ is a sequence of stepping-stone streams on the same intrusion path, then they will satisfy certain relation as defined below.

Definition 1 *A sequence of streams $(S_1, \ldots,\ S_{n+1})$ is a* normal sequence *if they are independent. It is a* stepping-stone sequence *if there exist bijections $g_i : \mathcal{T}_i \to \mathcal{T}_{i+1}$ ($i = 1, \ldots,\ n$) such that $g_i(s) - s \geq 0$ for all $s \in \mathcal{T}_i$.*

The bijection g_i is a mapping between the arrival and the departure times of packets at the ith host, allowing permutation of packets during the relay. The condition that g_i is a bijection imposes a *packet-conservation* constraint, *i.e.*, no packets are generated or dropped at the stepping stones. The condition $g_i(s) - s \geq 0$ is the *causality* constraint, which means that a packet cannot leave a host before it arrives.

For interactive stepping-stone attacks, there is usually a *maximum tolerable delay* for attacking packets, which is imposed by the physical constraints, the transmission protocol, or the need of the attacker. Stepping-stone sequences with maximum tolerable delays satisfy the following stronger definition.

Definition 2 *A sequence of streams $(S_1, \ldots,\ S_{n+1})$ is a* stepping-stone sequence with bounded delay *if it is a stepping-stone sequence, and there exists a constant $\Delta > 0$ such that for all $i = 1, \ldots,\ n$, $g_i(s) - s \leq \Delta$ for all $s \in \mathcal{T}_i$.*

The condition $g_i(s) - s \leq \Delta$ means that no attacking packets can stay at a stepping-stone host for longer than Δ.

If the attacker can insert chaff into his traffic, then the above constraints only apply to the fraction of his traffic which consists of real attacking packets, as stated in the following definition.

Definition 3 *A sequence of streams* $(S_1, \ldots, S_{n+1})$ *is a* stepping-stone sequence (with bounded delay) in chaff *if it is the superposition of a stepping-stone sequence (with bounded delay) and a sequence of chaff streams* $(C_1, \ldots, C_{n+1})$.

Stream C_i ($i = 1, \ldots, n+1$) consists of dummy packets called *chaff* which do not need to arrive at the victim. Chaff packets can be generated or dropped at any stepping stone hosts without affecting the attack. They are artificially inserted by the attacker to evade detection.

We consider centralized detection, where there is a central detector to test the following binary hypotheses:

$\mathcal{H}_0 : (S_1, \ldots, S_{n+1})$ is a normal sequence,
$\mathcal{H}_1 : (S_1, \ldots, S_{n+1})$ is a stepping-stone sequence,

by observing $(s_1^{(i)}, s_2^{(i)}, \ldots)_{i=1}^{n+1}$. In this paper, we consider stepping-stone attacks in which the attacker can perturb the timing subject to a bounded delay, and mix attacking packets with chaff packets.

3. FUNDAMENTAL LIMIT ON ONE-HOP STEPPING-STONE ATTACKS

In this section, we consider the simple case when $n = 1$, *i.e.,* there is only one stepping-stone host on the intrusion path. With enough chaff packets, the attacker can make his traffic look identical to any processes he wants. The problem is that the transmission of chaff packets causes a waste of rate. To launch attacks efficiently, the attacker will have the motivation to reduce the amount of chaff as much as possible.

Blum *et al.* in [8] propose an optimal chaff-inserting algorithm called "BOUNDED-GREEDY-MATCH" (BGM) which can embed a pair of stepping-stone streams with bounded delay into arbitrary point processes while inserting the minimum amount of chaff packets. Given a pair of incoming and outgoing streams at a host, BGM matches arrivals with departures subject to the constraints of causality and bounded delay. In [12], we combine the insertion of chaff and the transmission of attacking packets into the algorithm in Table 1. Then for each valid pair $(s_m^{(1)}, s_n^{(2)})$, the attacker can schedule an attacking packet to arrive at $s_m^{(1)}$ and depart at $s_n^{(2)}$.

Algorithm BGM has a low complexity of $O(|S_1|+|S_2|)$ because it only needs to scan (S_1, S_2) once and the amount of work in each iteration is constant. It is shown in [8] that BGM inserts the minimum chaff in embedding attacking packets with bounded delay into arbitrary point processes[1]. In [12], we characterize the minimum amount of chaff to mimic independent Poisson processes in the following theorem.

Table 1: BOUNDED-GREEDY-MATCH (BGM).

```
BOUNDED-GREEDY-MATCH(S_1, S_2, Δ):
  m = n = 1;
  while m ≤ |S_1| and n ≤ |S_2|
    if s_n^(2) − s_m^(1) < 0
      s_n^(2) = chaff; n = n + 1;
    else if s_n^(2) − s_m^(1) > Δ
        s_m^(1) = chaff; m = m + 1;
      else
        (s_m^(1), s_n^(2)) = a valid pair;
        m = m + 1; n = n + 1;
      end
    end
  end
```

Theorem 1 *If* S_1 *and* S_2 *are independent Poisson processes of equal rate* λ*, then BGM inserts* $1/(1+\lambda\Delta)$ *fraction of chaff among all the packets in* $S_1 \cup S_2$.

Remark: The theorem implies that the attacker can send attacking packets at rate $\lambda^2\Delta/(1+\lambda\Delta)$, whiling inserting chaff packets to make his traffic mimic independent Poisson processes of rate λ. For large λ, the attacker can send attacking traffic at rather high rate without possibly being detected by any activity-based detector.

4. FUNDAMENTAL LIMIT ON MULTI-HOP STEPPING-STONE ATTACKS

The result in Section 3 seems pessimistic in that it is possible that the detector has no way to detect encrypted one-hop stepping-stone attacks even if the attacker only transmits a small amount of chaff. It shows the weakness of detecting stepping-stone attacks on a local scale. If, however, the stepping-stone attack involves multiple hops, and there is a central detector which makes decisions based on the incoming and outgoing traffic at each hop, then the capability of the attacker to evade detection will be severely limited. We proceed by introducing a few definitions related to multi-hop stepping-stone attacks.

[1]The original proof in [8] is for independent binomial processes, but it holds for arbitrary processes.

Definition 4 *A* relay path *through a sequence of streams* $(S_1, \ldots, S_{n+1})$ *is a sequence of epochs from each of the streams* $(t_i \in S_i)_{i=1}^{n+1}$. *A relay path* $(t_1, \ldots, t_{n+1})$ *is* valid *for delay bound* Δ *if* $t_{i+1} - t_i \in [0, \Delta]$ *for all* $i = 1, \ldots, n$. *A set of relay paths is* feasible *if all the relay paths in it are disjoint and valid. A feasible set of relay paths is* order-preserving *if any two paths in it* $(t_i)_{i=1}^{n+1}$ *and* $(t'_i)_{i=1}^{n+1}$ *satisfy either* $t_i \le t'_i$ *for all* i *or* $t_i \ge t'_i$ *for all* i.

A valid relay path represents a sequence of timestamps at which an attacking packet is emitted from each of the stepping-stone hosts. To schedule the transmission of attacking packets, the attacker must find a feasible set of relay paths, and schedule the transmission of each attacking packet according to a different relay path. The requirement that paths in a feasible set are disjoint is because we do not allow the combining of multiple packets into a single relay packet. If a set of relay paths is order-preserving, then there will be no intersection between the paths, which greatly reduces the complexity in searching for a desired set of relay paths.

Proposition 1 *Among all the feasible sets of relay paths with the largest cardinality, there always exists a set which is order-preserving.*

Remark: By Proposition 1, we only need to search among order-preserving sets to find a largest feasible set of relay paths.

Proof: The proof is by direct observation. As illustrated in Fig. 1, suppose $(s_1^{(1)}, s_2^{(2)}, s_1^{(3)})$ and $(s_2^{(1)}, s_1^{(2)}, s_2^{(3)})$ are valid relay paths. By switching the intersected part, we obtain two order-preserving paths $(s_1^{(1)}, s_1^{(2)}, s_1^{(3)})$ and $(s_2^{(1)}, s_2^{(2)}, s_2^{(3)})$ which are also valid. We can restructure any largest feasible set of relay paths into an order-preserving set by repeatedly applying such switching.

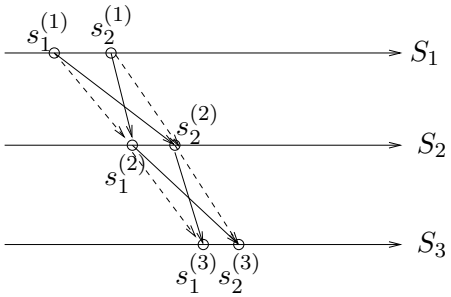

Fig. 1: Dashed lines denote alternative valid relay paths which preserve the order of incoming packets.

■

Given a sequence of streams $(S_i)_{i=1}^{n+1}$, suppose the attacker wants his traffic to mimic these streams. Then he wants to find the largest feasible set of relay paths so that he can transmit the maximum number of attacking packets. To this end, we derive an algorithm called "GREEDY-RELAY-EMBEDDING" (GRE) for finding the largest feasible set of relay paths. Algorithm GRE is presented in Table 2.

Table 2: GREEDY-RELAY-EMBEDDING (GRE).

GREEDY-RELAY-EMBEDDING$(S_1, \ldots, S_{n+1}, \Delta)$:

```
for j = 1 : |S_{n+1}|
  C_{n+1,j} = {s_j^{(n+1)}};
  for i = n : -1 : 1
    for all s ∈ S_i ∩ [s_j^{(n+1)} − (n − i + 1)Δ, s_j^{(n+1)}]
      if (s is unselected) and ([s, s + Δ] ∩ C_{i+1,j} ≠ ∅)
        add s to C_{i,j};
        s.next = min([s, s + Δ] ∩ C_{i+1,j});
      end
    end
  end
  if |C_{1,j}| ≠ 0
    select s_{m_1}^{(1)} = min(C_{1,j});
    for i = 2 : n + 1
      select s_{m_i}^{(i)} = s_{m_{i−1}}^{(i−1)}.next;
    end
    (s_{m_i}^{(i)})_{i=1}^{n+1} is a valid relay path;
  end
end
```

The complexity of GRE is $O(n^3|S_{n+1}|)$, or more precisely, about $\frac{1}{3}(\lambda\Delta)^2 n^3 |S_{n+1}|$ on the average[2], where λ is the maximum rate of $S_1, \ldots, S_n$. The set $C_{i,j}$ in GRE is the set of all possible predecessors in S_i of the jth point in S_{n+1}, *i.e.,*

$$C_{i,j} = \{t \in S_i : t \text{ is unselected, and } \exists \text{ a valid relay path of unselected points from } t \text{ to } s_j^{(n+1)}\}.$$

Algorithm GRE is based on the idea that among all the valid relay paths for a particular incoming packet, we should choose the earliest one to maximally avoid conflicting with the following incoming packets. For each departing packet from the last host $s_j^{(n+1)} \in S_{n+1}$, GRE recursively find the

[2]The dominating step is the recursive computation of $C_{i,j}$'s. There are at most $(n-i+1)\lambda\Delta$ points in S_i on the average which are possible to join $C_{i,j}$, and for each of these points, GRE needs no more than $(n-i)\lambda\Delta$ steps to check the condition $[s, s+\Delta] \cap C_{i+1,j} \neq \emptyset$; GRE needs up to $(n-i)(n-i+1)(\lambda\Delta)^2$ steps to compute $C_{i,j}$. The total complexity is then calculated as $|S_{n+1}| \sum_{i=1}^{n} (n-i)(n-i+1)(\lambda\Delta)^2 \approx \frac{1}{3}(\lambda\Delta)^2 n^3 |S_{n+1}|$.

sets $\{C_{i,\,j}\}_{i=n}^{1}$ of all its possible predecessors in each of the streams $S_n, \ldots, S_1$. The construction of $C_{i,\,j}$ makes sure that every point in it has a valid relay path to $s_j^{(n+1)}$, and this path will not conflict with paths that are already selected to relay packets before $s_j^{(n+1)}$. If $C_{1,\,j}$ is not empty, then there must be a valid path from some incoming point in S_1 to $s_j^{(n+1)}$, and GRE selects the earliest of them.

After GRE finds a set of relay paths, the attacker can schedule the transmission of attacking packets accordingly. The unselected points will be the transmission times of chaff packets. It is easy to see that the set of relay paths found by GRE is feasible. The optimality of GRE is guaranteed by the following proposition.

Proposition 2 *Given a realization of point processes $(S_i)_{i=1}^{n+1}$, GRE finds the largest feasible set of relay paths from S_1 to S_{n+1}.*

Proof: See Appendix. ■

Since GRE is optimal in the sense that it requires the transmission of the minimum number of chaff packets, the performance of GRE gives fundamental limits to the attacker's capability of sending attacking packets. By analyzing GRE, we bound the attacker's ability to send attacking traffic while keeping his traffic completely undetectable to activity-based detectors by adding chaff, as stated in the following theorem.

Theorem 2 *Suppose the attacker wants all the streams on the intrusion path to mimic independent Poisson processes with equal rate λ. Then for an intrusion path of length n, the rate of attacking traffic is upper bounded by $\lambda(1 - e^{-\lambda\Delta})^n$.*

Proof: See Appendix. ■

Remark: Theorem 2 says that if the attacker wants to completely hide the intrusion path, the rate of attacking traffic decays exponentially with the increase in the length of the intrusion path. This result guarantees that the attacker's capability of launching attacks is severely constrained by the number of hops he takes in the chain of stepping stones. To send attacking commands at a sufficiently high rate, The attacker has to either leave some connections on the intrusion path correlated, or reduce the number of stepping stones on the intrusion paths, both of which makes the attacker vulnerable to detection and tracing.

5. RANDOMIZING PACKET SCHEDULING TO DEFEND AGAINST STEPPING-STONE ATTACKS

In Section 4, we have established a fundamental limit on the rate of attacking traffic through multiple stepping stones. The result requires that the attacker wants his traffic to mimic Poisson processes. Although we can not control the attacker's decision, as the network designer, we can force the attacker to choose Poisson processes by scheduling other traffic as Poisson. Suppose normal traffic all consists of Poisson processes. Then we can install local detectors at the hosts to test whether the interarrival distribution is exponential; all traffic with non-exponential interarrival distributions will be considered abnormal. Next, a global detector can test the dependency among connections to detect stepping-stone traffic. The global detection can be done either in a centralized fashion at a fusion center, or in a distributed fashion by conferencing among local detectors. Using this framework, we show that, at least in principle, scheduling packet transmissions as Poisson processes allows us to restrain the efficiency of stepping-stone attacks.

We note that the key to impeding stepping-stone attacks is to randomize packet transmissions. Due to the randomization, traffic flows can be traced without using their content because every flow will have unique timing characteristics which allow us to distinct it from all the other flows. On the other hand, scheduling schemes not involving randomization are vulnerable to stepping-stone attacks due to the lack of uniqueness. We illustrate this idea by the following comparison.

We compare a randomized scheduling scheme with a deterministic scheme. In the randomized scheduling, we assume that packets are transmitted according to Poisson processes. In the deterministic scheduling, we assume that packets are transmitted according to deterministic point processes with constant interarrival time D, for some constant $D > 0$. Assume that for a pair of independent deterministic processes, the difference between an arrival and the first departure after the arrival is uniformly distributed in $[0,\, D)$.

Suppose the attacker has a maximum tolerable delay Δ, and he wants to transmit attacking packets through n stepping stones without being detected. In a network using the randomized scheduling, the attacker can make his traffic undetectable by transmitting attacking packets together with chaff according to independent Poisson processes. For fair comparison, we let these Poisson processes have equal rate $\lambda = 1/D$. By Theorem 2, we can bound the rate of attacking traffic λ_R as

$$\lambda_R \leq \lambda(1 - e^{-\lambda\Delta})^n, \quad \text{for all } \lambda\Delta.$$

In a network using the deterministic scheduling, chaff does not help the attacker in evading detection because given the realization of a pair of deterministic processes, he either can transmit attacking packets at all the arrival epochs, or cannot transmit any attacking packets, depending on whether

the difference between an arrival and the first following departure is bounded by Δ or not. Then it is easy to see that, if $\Delta \geq D$, the attacker can delay attacking packets by a time randomly chosen from $[0,\ D)$ at every stepping stone, and there is no way to detect the attacking traffic. If $\Delta < D$, then the detector can simply declare a sequence of flows as attacking traffic if all the delays are bounded by Δ, and the attacker cannot evade the detection. Such a detector will have a lot of false alarms for one-hop transmissions, but as the number of hops increases, the false alarm probability will decay exponentially as $(\lambda\Delta)^n$.

For delay $\Delta \geq D$, we see that in both scheduling strategies it is possible for the attacker to completely evade detection; the rate of attacking traffic will, however, decay exponentially with the increase of the number of stepping stones in the randomized scheduling, but stay constant in the deterministic scheduling. If $\Delta < D$, there is a detector in the deterministic scheduling that cannot be evaded. The feasibility of such a deterministic scheduling strategy is, however, problematic. We can expect that in a network with deterministic scheduling, a lot of transmissions will be dummy packets because nodes may not have packets at the scheduled transmission times; furthermore, bursty traffic will not be supported well since the rate is fixed at $1/D$. From this point of view, the randomized scheduling gives the network more flexibility.

6. CONCLUSION

In this paper, we show that in principle randomization in packet transmissions facilitates the defense against stepping-stone attacks. The drawback is that such randomization may be undesirable in certain applications such as interactive sessions or audio transmissions where the data are time sensitive.[3]

7. APPENDIX

7.1. Proof of Proposition 2

By Proposition 1, it suffices to show that GRE finds the largest set of relay paths among all the feasible sets of relay paths that preserve the order of incoming packets.

Let $\mathcal{P}$ be the set of relay paths found by GRE, and $\mathcal{P}^*$ a largest feasible set of relay paths that is order-preserving. Suppose $s_1 \in S_{n+1}$ is the endpoint of a relay path $p_1^* \in \mathcal{P}^*$, as illustrated in Fig. 2, but there is no relay path in $\mathcal{P}$ leading to s_1. Then in $\mathcal{P}$ there must be relay path(s) having some overlap with p_1^*, and leading to point(s) in S_{n+1} before s_1; otherwise, GRE would have chosen p_1^* or some path no later than p_1^* to lead to s_1. Let the latest of these points be s_2, and its path in $\mathcal{P}$ be p_1. If s_2 does not correspond to any relay path in $\mathcal{P}^*$, we stop tracing; otherwise, let $p_2^* \in \mathcal{P}^*$ lead to s_2. We know that there have to be relay path(s) in $\mathcal{P}$ partly overlapping with p_2^*; if not, GRE would have chosen a path no later than p_2^* to lead to s_2, but this path would not have overlap with p_1^*, which is a contradiction. We continue tracing by alternately choosing the latest path p_i in $\mathcal{P}$ which has partial overlap with p_i^*, and then finding a path $p_{i+1}^* \in \mathcal{P}^*$ with the same endpoint as p_i for $i = 2,\ 3, \ldots$. The tracing continues until we find a point which has a relay path in $\mathcal{P}$ but not $\mathcal{P}^*$, or we reach a relay path p_m in $\mathcal{P}$ leading to a point s_{m+1} which is before the endpoint s_m of the first relay path p_m^* in $\mathcal{P}^*$.

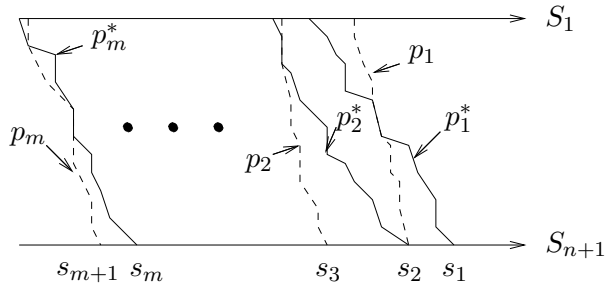

Fig. 2: Every relay path in $\mathcal{P}^*$ corresponds to a path in $\mathcal{P}$; solid line: paths in $\mathcal{P}^*$; dashed line: paths in $\mathcal{P}$.

Therefore, we see that every relay path in $\mathcal{P}^*$ corresponds to a relay path in $\mathcal{P}$. This proves that $\mathcal{P}$ is also a largest feasible set of relay paths.

■

7.2. Proof of Theorem 2

We bound the rate of attacking traffic by obtaining an upper bound on the asymptotic fraction of attacking packets in S_1. We first show that this fraction is upper bounded by the probability that the first incoming packet can be an attacking packet, and then bounded this probability.

Be Proposition 2, it suffices to bound the fraction of attacking packets scheduled by GRE. For an incoming packet $s_k^{(1)}$ ($k \geq 2$), given a feasible and order-preserving set of relay paths for incoming packets before $s_k^{(1)}$ found by GRE, the conditional probability for $s_k^{(1)}$ to have a valid relay path is equal to

$$\Pr\{\exists (t_i \in S_i)_{i=2}^{n+1},\ t_i \in [\max(t_{i-1},\ t_i'),\ t_{i-1} + \Delta]\},$$

where $t_1 = s_k^{(1)}$, and t_i' is the latest point in S_i which has been selected by GRE. The condition $t_i \geq t_i'$ represents

[3]The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U. S. Government.

the order-preserving requirement. We can easily bound this probability from above by

$$\Pr\{\exists (t_i \in S_i)_{i=2}^{n+1},\ t_i \in [t_{i-1},\ t_{i-1}+\Delta]\},$$

which is equal to the probability that $s_1^{(1)}$ has a valid relay path.

Next we prove by induction that the probability for $s_1^{(1)}$ to have a valid relay path of length n is equal to $(1-e^{-\lambda\Delta})^n$. Let $t_1 = s_1^{(1)}$. For $n = 1$, we have

$$\Pr\{\exists t_2 \in S_2,\ t_2 \in [t_1,\ t_1+\Delta]\} = 1 - e^{-\lambda\Delta}.$$

Assume that the result holds for relay path of length $n-1$ ($n \geq 2$). Then we have

$$\begin{aligned}
&\Pr\{\exists (t_i \in S_i)_{i=2}^{n+1},\ t_i \in [t_{i-1},\ t_{i-1}+\Delta]\} \\
&= \int_0^{\Delta} \lambda e^{-\lambda x} \Pr\{\exists (t_i \in S_i)_{i=2}^{n+1},\ t_i \in [t_{i-1},\ t_{i-1}+\Delta] \\
&\qquad\qquad \big| t_2 - t_1 = x\} dx \\
&= \int_0^{\Delta} \lambda e^{-\lambda x} (1-e^{-\lambda\Delta})^{n-1} dx \qquad (1) \\
&= (1-e^{-\lambda\Delta})^n,
\end{aligned}$$

where we use the induction assumption in (1).

Combining the facts that S_1 has rate λ, and at most $(1-e^{-\lambda\Delta})^n$ fraction of the packets are attacking packets, we conclude that the rate of attacking traffic is upper bounded by $\lambda(1-e^{-\lambda\Delta})^n$.

■

8. REFERENCES

[1] S. Staniford-Chen and L. Heberlein, "Holding intruders accountable on the internet," in *Proc. the 1995 IEEE Symposium on Security and Privacy*, (Oakland, CA), pp. 39–49, May 1995.

[2] X. Wang, D. Reeves, S. Wu, and J. Yuill, "Sleepy watermark tracing: An active network-based intrusion response framework," in *Proc. of the 16th International Information Security Conference*, pp. 369–384, 2001.

[3] Y. Zhang and V. Paxson, "Detecting stepping stones," in *Proc. the 9th USENIX Security Symposium*, pp. 171–184, August 2000.

[4] K. Yoda and H. Etoh, "Finding a connection chain for tracing intruders," in *6th European Symposium on Research in Computer Security, Lecture Notes in Computer Science 1895*, (Toulouse, France), October 2000.

[5] X. Wang, D. Reeves, and S. Wu, "Inter-packet delay-based correlation for tracing encrypted connections through stepping stones," in *7th European Symposium on Research in Computer Security, Lecture Notes in Computer Science 2502*, pp. 244–263, 2002.

[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 *5th International Symposium on Recent Advances in Intrusion Detection, Lecture Notes in Computer Science 2516*, 2002.

[7] X. Wang and D. Reeves, "Robust correlation of encrypted attack traffic through stepping stones by manipulation of inter-packet delays," in *Proc. of the 2003 ACM Conference on Computer and Communications Security*, pp. 20–29, 2003.

[8] A. Blum, D. Song, and S. Venkataraman, "Detection of Interactive Stepping Stones: Algorithms and Confidence Bounds," in *Conference of Recent Advance in Intrusion Detection (RAID)*, (Sophia Antipolis, French Riviera, France), September 2004.

[9] T. He and L. Tong, "A Signal Processing Perspective to Stepping-stone Detection," in *Proc. 2006 Conference on Information Sciences and Systems*, (Princeton, NJ), March 2006.

[10] P. Peng, P. Ning, D. Reeves, and X. Wang, "Active Timing-Based Correlation of Perturbed Traffic Flows with Chaff Packets," in *Proc. 25th IEEE International Conference on Distributed Computing Systems Workshops*, (Columbus, OH), pp. 107–113, June 2005.

[11] L. Zhang, A. Persaud, A. Johson, and Y. Guan, "Detection of Stepping Stone Attack under Delay and Chaff Perturbations," in *Proc. of the 25th IEEE International Performance Computing and Communications Conference (IPCCC 2006)*, (Phoenix, AZ), April 2006.

[12] T. He and L. Tong, "Detecting Stepping-stone Traffic in Chaff: Fundamental Limits and Robust Algorithms," Tech. Rep. ACSP-TR-06-06-01, Cornell University, June 2006. `http://acsp.ece.cornell.edu/pubR.html`.