University of Dayton
eCommons

Computer Science Faculty Publications | Department of Computer Science

3-2012

Stochastic Analysis of Horizontal IP Scanning

Derek Leonard
Texas A & M University - College Station

Zhongmei Yao
University of Dayton, zyao01@udayton.edu

Xiaoming Wang
Texas A & M University - College Station

Dmitri Loguinov
Texas A & M University - College Station

Follow this and additional works at: http://ecommons.udayton.edu/cps_fac_pub


Part of the Computer Security Commons, Databases and Information Systems Commons, OS and Networks Commons, Other Computer Sciences Commons, Systems Architecture Commons, and the Theory and Algorithms Commons

eCommons Citation

Leonard, Derek; Yao, Zhongmei; Wang, Xiaoming; and Loguinov, Dmitri, "Stochastic Analysis of Horizontal IP Scanning" (2012). *Computer Science Faculty Publications.* Paper 3.
http://ecommons.udayton.edu/cps_fac_pub/3

This Conference Paper is brought to you for free and open access by the Department of Computer Science at eCommons. It has been accepted for inclusion in Computer Science Faculty Publications by an authorized administrator of eCommons. For more information, please contact frice1@udayton.edu, mschlangen1@udayton.edu.

Stochastic Analysis of Horizontal IP Scanning

Derek Leonard, Zhongmei Yao, Xiaoming Wang, and Dmitri Loguinov*
Department of Computer Science and Engineering
Texas A&M University, College Station, TX 77843 USA
Email: {dleonard, mayyao, xmwang, dmitri}@cse.tamu.edu

***Abstract*—Intrusion Detection Systems (IDS) have become ubiquitous in the defense against virus outbreaks, malicious exploits of OS vulnerabilities, and botnet proliferation. As attackers frequently rely on host scanning for reconnaissance leading to penetration, IDS is often tasked with detecting scans and preventing them. However, it is currently unknown how likely an IDS is to detect a given Internet-wide scan pattern and whether there exist sufficiently fast scan techniques that can remain virtually undetectable at large-scale. To address these questions, we propose a simple analytical model for the window-expiration rules of popular IDS tools (i.e., Snort and Bro) and utilize a variation of the Chen-Stein theorem to derive the probability that they detect some of the commonly used scan permutations. Using this analysis, we also prove the existence of stealth-optimal scan patterns, examine their performance, and contrast it with that of well-known techniques.**

I. Introduction

As the Internet has grown more hostile over time [26], [38], many networks now deploy Intrusion Detection Systems (IDS) [5], [35] to deal with the constant pressure of unsolicited traffic and attempts to exploit various vulnerabilities at end-hosts [26]. In its most general form, IDS monitors all inbound/outbound connections to detect such activities as *scanning* (e.g., attempts to find open services [1], [10], [26], [37], [41]), *intrusion* (e.g., malicious packets that exploit known vulnerabilities [21], [23], [33]), *anomalies* (e.g., new communication patterns indicating infection [7], [14], [36]), and *DoS attacks* (i.e., suspicious spikes in traffic/connection volume [15], [22]). In conjunction with firewalls, IDS can block offending hosts and raise alarms to alert administrators to potentially undesirable activity.

To maintain scalability [17], adapt over time, and keep state from growing to infinity, existing IDS tools [5], [11], [24], [29], [35] utilize *window-based* processing of incoming traffic, which entails keeping per-flow statistics only for a limited period of time and applying IDS detection algorithms to the packets accumulated during this window. This makes the IDS detection process purely regenerative [31] and oblivious to any attacks that span multiple windows. One activity whose detection is particularly sensitive to the amount of state in each window is *horizontal scanning*, which consists of probing every Internet host on a given port to see if it is visible outside the firewall.

To balance accuracy and false-positive rates, an IDS typically requires some minimum number of packets in the window before triggering an estimator or raising an alarm. As observed in [38], a worm could utilize so-called *stealthy* scan patterns to prevent IDS from reaching this threshold, which makes such scans equally powerful against all underlying estimators. For horizontal stealth scanning studied in this paper, the main exposed technique [38] is to scan "very slowly," potentially dragging out the process over several months. However, it is unclear whether stealth scanning is possible at faster rates, in what particular order the IP space should be probed, and how likely the existing IDS packages are to detect such approaches. To shed light on this issue, we model window rules of two popular IDS implementations (i.e., Snort [35] and Bro [5]), study the rates at which the existing scan techniques [1], [9], [19], [20], [27], [28], [38] become stealthy, and explore fundamental IDS limitations under stealth-optimal scan patterns.

While IDS avoidance in the literature commonly targets vulnerabilities of known implementations [12], [13], [25], [30], [34] or concealment of abnormal communication patterns [6], [39], [43], to our knowledge the performance of *generic* window-based IDS and various scan techniques has not been modeled before. We perform this task below.

II. Formalizing Scanning

In this section, we outline the goals of a large-scale scanner, introduce three fundamental elements of a scan that determine its performance, and set forth assumptions on the various types of IDS. We then discuss stealth-optimal scans and their properties.

A. Scan Objectives

Assume $\mathcal{F} = \{0, 1, \ldots, n\}$ is the IPv4 address space, where $n = 2^{32}$, and $\mathcal{S}$ is the set of all CIDR networks. As discussed in [38], one of the most effective penetration models used by an attacker (i.e., the Flash worm) relies on a two-phase scan/infect approach. The first phase scans $\mathcal{F}$ using m source IPs in some set $\mathcal{M}$ (e.g., a subset of the attacker's botnet) to build a list of vulnerable targets $\mathcal{V}$. The second phase uses zombie hosts in another set $\mathcal{M}'$ to attempt infection of $\mathcal{V}$ using a new exploit. Sets $\mathcal{M}$ and $\mathcal{M}'$ may overlap if exposure during the first phase does not reduce the infection performance of IPs in $\mathcal{M}$ during the second phase.

As there is no need for newly infected hosts to scan the entire Internet, they perform only a limited scan of the local network (e.g., the corresponding BGP prefix, which is assumed

*Supported by NSF grant CNS-1017766.

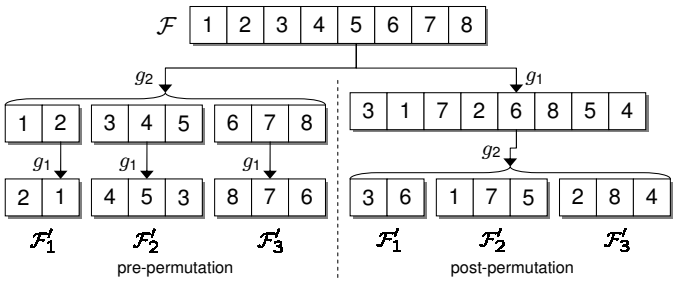


Fig. 1. Illustration of permutation/split ($m = 3$).

to be inside the firewall) and then stop. Due to the short duration of the infection phase (hours rather than weeks) and limited local scanning, this attack is difficult to stop once it starts and infections are hard to detect after phase two is over.

For a given budget m and fixed scan duration T, we assume the attacker's goal is to minimize its detection probability at each CIDR subnet s (i.e., maximize its stealthiness) during the first phase of the attack. The problem of delivering malicious payload is implementation/exploit-dependent and outside the scope of this paper. Due to the static nature of set $\mathcal{M}$, we are also not concerned with sub-allocating the scan space dynamically to each newly infected host as commonly studied in worm propagation [20].

B. Scan Patterns

Our first contribution is systematic classification of the algorithms involved in scanning. Define an Internet-wide *scan pattern* to consist of three principle elements – permutation, split, and schedule. The existing literature [3], [4], [8], [21], [23], [27], [28], [33] has glanced over the first two elements, but without any formalization or analysis. The third one is novel and is presented here for the first time.

Given a list of items $\mathcal{F}$, a *permutation* is a one-to-one mapping function $g_1 : \mathcal{F} \to \{1, 2, \ldots, |\mathcal{F}|\}$ that simply shuffles the elements in $\mathcal{F}$. We often denote the permuted sequence by $\mathcal{F}' = g_1(\mathcal{F})$. Permuting the IP space is highly beneficial because it reduces the instantaneous load on target networks, increases delays between packets entering IDS, and generally lowers the detection probability. It can also control randomness and correlation among the destinations in each s.

We define a *split* as a many-to-one function $g_2 : \mathcal{F} \to \mathcal{M}$ that assigns the elements of list $\mathcal{F}$ to scanner IPs. One can view this as a partition of $\mathcal{F}$ into non-overlapping lists $\mathcal{F}_1, \ldots, \mathcal{F}_m$, where $\mathcal{F}_i$ is given to host $i \in \mathcal{M}$. If each of $\mathcal{F}_i$ is an ordered subset of $\mathcal{F}$, we call this arrangement a *block-split*. In the context of the Internet, a *pre-permutation* scanner [4], [25] first applies partitioning g_2 to $\mathcal{F}$ and then permutes each $\mathcal{F}_i$ using some algorithm g_1 to produce the final assignment $\mathcal{F}'_i = g_1(\mathcal{F}_i)$ of source i. A *post-permutation* scanner [3], [8], [18], [19], [20], [28], [42], [44] first applies permutation g_1 to $\mathcal{F}$ and then partitions list $\mathcal{F}'$ using g_2 into $\mathcal{F}'_1, \ldots, \mathcal{F}'_m$. This is schematically shown in Fig. 1, where the pre-permutation scanner (left side) uses a block-split, while the post-permutation one (right side) does not.

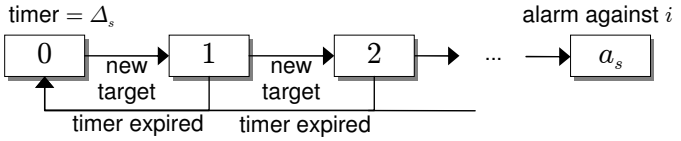


Fig. 2. Process $C_i^s(t)$ of IDS-A.

The final issue is to determine how each host i probes its target set $\mathcal{F}'_i$ so as to complete the scan by a certain time T. To allow i to periodically send packets faster or slower than its average rate $r_i = |\mathcal{F}'_i|/T$, define a *schedule* to be a many-to-one function $g_3 : \mathcal{F}'_i \to [0, T]$ that decides the exact time instances at which i hits each of its assigned targets. While all existing scanners draw elements from $\mathcal{F}'_i$ with a constant inter-probe delay $1/r_i$, bursty patterns will be discussed shortly.

C. Window-Based IDS

To understand the relationship between detectability of a scan and its probing rate r, one requires a model of IDS. In what follows, we present our second contribution that consists of formalized window-based detection rules of popular IDS packages [5], [11], [24], [35] and firewall-log analyzers [29]. Since scalability requires that IDS expire state and operate in windows of finite size [17], other high-performance IDS designs are also likely to fall under one of the two categories introduced here.

Our first model, which we call IDS-A, stems from the rules of Snort [35] and its commercial implementations [11], [24]. For each source IP $i \in \mathcal{M}$ sending packets into a given subnet $s \in \mathcal{S}$ protected by an IDS, define $C_i^s(t)$ to be the count of unique targets seen by the IDS from i in the interval $[0, t]$. Since keeping infinite history of hosts contacted by i incurs substantial RAM/CPU overhead and fails to properly discount outdated information, IDS-A periodically resets i's state back to zero as illustrated in Fig. 2. Here, random process $C_i^s(t)$ increases by 1 for each new target hit by i, returns to state 0 every Δ_s time units, and absorbs in pre-defined threshold state $a_s \geq 1$ that triggers some internal estimation algorithm, which we assume *always detects the scanner once invoked*[1].

Our second model, which we call IDS-B, is derived from the techniques used by Bro [5] and certain firewall-log analyzers [29]. In this method, $C_i^s(t)$ represents the number of unique unresponsive targets hit by i in the interval $[0, t]$. Unlike IDS-A, this model expires i's state *only* if it does not probe any new unresponsive targets for Δ_s time units. Assuming the worst-case scenario where *none of the targets respond*, this logic can be described by Fig. 3, where the expiration timer of i resets to Δ_s upon each state transition. This represents the best-case detection scenario for the estimator (e.g., TRW [10], CBCRL [32]) that runs on top of the underlying packet-capture device.

For the same parameter set, IDS-B is stricter than IDS-A in the sense that any scanner detected by the latter is

[1]Note that deriving the probability that the triggered estimator detects the scan depends on numerous factors (e.g., fraction of internal IPs with legitimate servers, prior history of inbound/outbound traffic, and specific detection algorithms) and is outside the scope of the paper.

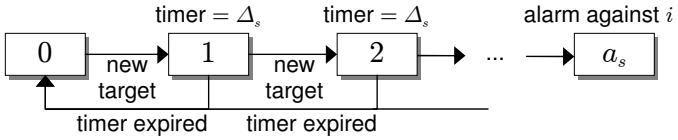


Fig. 3. Process $C_i^s(t)$ of IDS-B.

always detected by the former. Similarly, a scanner avoiding IDS-B always avoids IDS-A. However, IDS-B achieves this improvement at the expense of maintaining a separate timer for each i and stochastically higher overhead (i.e., longer lists of seen targets) in steady-state. Default parameters (Δ_s, a_s) of deployed open-source and commercial IDS-A/B are summarized in Table I.

D. Stealth

Our next contribution is to introduce the concepts of detectability and stealthiness of a scan. Let $\mathcal{I} \subseteq \mathcal{S}$ be the set of all IDS-equipped networks, where each element of $\mathcal{I}$ is a full CIDR block (often written in the $/x$ notation). Then, we start with the following classification.

Definition 1: A network $s \in \mathcal{I}$ is called *size-trivial* if $m(a_s - 1) \geq |s|$, *unavoidable* if $a_s = 1$, and *normal* otherwise.

Size-trivial subnets can be covered with fewer than a_s packets per source IP, which means they pose no threat of detection if the scanner can probe them while perfectly load-balancing between its IPs in $\mathcal{M}$. In contrast, unavoidable networks raise an alarm on the very first probe (e.g., darknets, personal firewalls) and thus cannot be avoided in practice by any scanner. Define $\mathcal{I}_{ST}, \mathcal{I}_U, \mathcal{I}_N$ to be pair-wise non-overlapping sets of respectively size-trivial, unavoidable, and normal networks in $\mathcal{I}$.

Assume $r = n/T$ is the average scanning rate. Then, for each source IP $i \in \mathcal{M}$, let

$$\tau_i^s(r) = \inf\{t > 0 : C_i^s(t) = a_s | C_i^s(0) = 1\} \qquad (1)$$

be the amount of time it takes s to detect i, which is simply the first hitting time of $C_i^s(t)$ onto state a_s after the IDS sees the initial packet from i. Let $A_i^s(r)$ be an indicator variable of detection event $\tau_i^s(r) < T$ and $A^s(r) = \sum_{i \in \mathcal{M}} A_i^s(r)$ be the number of source IPs detected by subnet $s \in \mathcal{I}$ in $[0, T]$. Then, $\rho^s(r) = P(A^s(r) \geq 1)$ is the probability that network s detects the scan at rate r.

Assume X is a pattern that scans all IPs in $\mathcal{F}$. Then, define the *stealth-cover time* (SCT) T_X^s of a normal subnet $s \in \mathcal{I}_N$ to be the minimum scan duration T that allows X to avoid detection at s. Recalling that $r = n/T$, observe that $T_X^s = \inf\{t \geq 0 : \rho^s(n/t) = 0\}$. Note that the concept of SCT applies only to normal subnets since size-trivial networks can be scanned without detection in $T_X^s = 0$ and unavoidable networks require $T_X^s = \infty$, neither of which is helpful in establishing the performance of scanning algorithms.

Definition 2: A scan pattern X is called *k-stealthier* in $s \in \mathcal{I}_N$ than Y if it exhibits k times smaller SCT, i.e., $T_X^s = T_Y^s/k$. It is called *IP-scalable* if it is m-stealthier in all $s \in \mathcal{I}_N$ with m source IPs than with one.

TABLE I
PARAMETERS OF COMMON IDS

Type	Name	Δ_s (sec)	a_s
IDS-A	Snort [35]	60	5
	Juniper [11]	120	50
	NIKSUN [24]	300	200
IDS-B	Bro [5]	600	20
	Bro TRW [10]	1800	4
	Psad [29]	3600	5

The concept of k-stealthier is used later in the paper to compare the relative performance of different scan patterns. IP-scalability, on the other hand, determines whether a particular scan pattern can reduce its scan duration T proportional to the number of participating IPs without becoming more detectable. Interestingly, some of the methods discussed below do not benefit from larger m and are not IP-scalable.

Our final definition relates to stealth optimality. It is usually safe to assume that the scanner remains oblivious to individual IDS values (Δ_s, a_s) and CIDR subnet boundaries in set $\mathcal{I}$. However, from the analysis of common IDS implementations (e.g., Bro-TRW [10] requires at least 4 samples for its estimator), one may possess a uniform lower bound β on parameter a_s. In that case, we call the scanner *β-aware* if $2 \leq \beta \leq a_s$ holds simultaneously for all normal subnets $s \in \mathcal{I}_N$ and no larger bound is known. If $\beta = 2$, we call the algorithm *unaware* since it benefits from no additional knowledge.

Definition 3: For a given m, a β-aware scan pattern X is called *STealth-OPtimal* (STOP) if for both IDS-A/B it 1) achieves $\rho^s(r) = 0$ in all size-trivial networks; and 2) minimizes the SCT of all normal subnets among all β-aware patterns, i.e., $\forall s \in \mathcal{I}_N : T_X^s = \min_Y T_Y^s$.

III. ANALYSIS OF EXISTING METHODS

Our goal in this section is to analyze two popular methods for scanning the Internet – sequential [1], [9], [19] and uniform [20], [27], [28], [38]. Our contribution here is not only to derive the detection probability $\rho^s(r)$ and cover time T_X^s for both IDS-A/B, but also to develop a novel unifying modeling framework that applies to both pre and post-permutation splits.

A. Sequential

Our first studied method, which we call *sequential*, does not permute the IP space (i.e., $\mathcal{F}' = \mathcal{F}$), uses a block-split that partitions $\mathcal{F}$ into m equal-size chunks, and sends packets from each i with constant spacing $\delta = 1/r_i = Tm/n$. Note that both pre/post permutation splits are equivalent for this method and each subnet s (smaller in size than n/m and not falling on the boundary between adjacent source IPs) is scanned by a single $i \in \mathcal{M}$ assigned to it.

The sequential permutation is guaranteed to avoid IDS-A if and only if each source allows no more than $\beta - 1$ inter-packet gaps within any interval $[t, t + \Delta_s)$, which is equivalent to $\delta(\beta - 1) \geq \Delta_s$. For IDS-B, this condition is much more conservative since none of the inter-packet delays δ can be smaller than Δ_s. Combining the two cases, we have

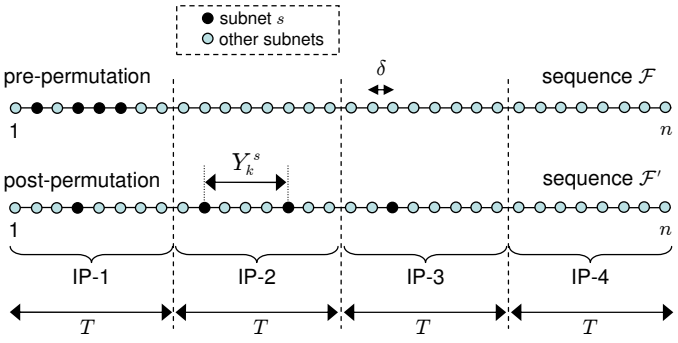


Fig. 4. Uniform model ($m = 4, |s| = 4$).

the sequential SCT as

$$T_Q^s = \frac{\Delta_s n}{m\zeta}, \quad \text{where} \quad \zeta = \begin{cases} \beta - 1 & \text{IDS-A} \\ 1 & \text{IDS-B} \end{cases}. \tag{2}$$

Notice from (2) that sequential scanning is IP-scalable and $(\beta - 1)$-stealthier against IDS-A than IDS-B. This pattern is invisible to all networks with fewer hosts than the IDS threshold (i.e., $|s| < a_s$), but this is far from optimal since IP-load-balancing can do much better, i.e., automatically avoid detection at all size-trivial networks whose $|s| \le m(a_s - 1)$. This difference is quite significant for large m.

In terms of probing rates, sequential scans hit each s at

$$\max\Big(\frac{n}{mT}, \frac{|s|}{T}\Big) \tag{3}$$

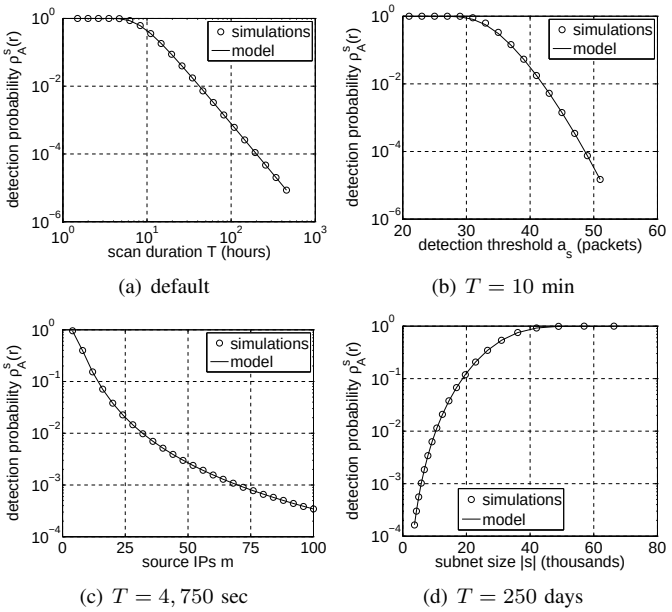
packets per second (pps). Depending on the scan duration T, this rate may become quite noticeable in comparison to the background traffic and may lead to easy detection. For $T = 24$ hours, the first term of (3) is $49.7/m$ Kpps, regardless of the target subnet size. However, if both the botnet and s are large (i.e., $m|s| \approx n$), the sequential scan rate might not be too far from the optimal $|s|/T$, which is possibly one of the reasons for its widespread use in the Internet [1].

B. Uniform Pattern

The main drawback to the sequential permutation is that it does not explore other subnets before hitting the same s with repeat packets. Uniform scanning improves upon this basic algorithm by spreading packets between random subsets of the Internet. We call a permutation function g_1 on list $\mathcal{F}$ *uniform* if the probability that each $i \in \mathcal{F}$ moves into position $j \in [1, |\mathcal{F}|]$ is $1/|\mathcal{F}|$. All existing uniform scanners use block-split and constant inter-packet delays $\delta = Tm/n$.

Consider a particular subnet s with $|s|$ IPs that need to be scanned in $[0, T]$. For pre-permutation, the uniform shuffle randomly scatters these $|s|$ targets throughout $\mathcal{F}_i'$, where i is the host permanently assigned to scanning s and $\mathcal{F}_i'$ is its list of targets. For post-permutation, the same IPs are now scattered in a much larger set $\mathcal{F}'$. This is illustrated in Fig. 4, where the IPs in s are marked with solid circles.

Assuming $n \gg 1$, the shuffle can be viewed as occurring in *time* rather than inside a discrete set. This transformation



(a) default (b) $T = 10$ min

(c) $T = 4,750$ sec (d) $T = 250$ days

Fig. 5. Comparison of post-permutation IDS-A model (5) to simulations (default parameters $|s| = 2^8, \Delta_s = 60$ sec, $a_s = 4$, and $m = 1$).

simplifies understanding of the derivations below and does not impact the accuracy of analysis. Specifically, imagine that source IPs scan the Internet *one after the other* (rather than concurrently) as shown at the bottom of Fig. 4 with four intervals of size T following each other back-to-back. This can be done because IDS does not correlate traffic from different source IPs. Then, the time instances when s sees probes from $\mathcal{M}$ is distributed uniformly in the interval $[0, \omega T]$, where

$$\omega = \begin{cases} 1 & \text{pre-permutation} \\ m & \text{post-permutation} \end{cases}. \tag{4}$$

C. Uniform Against IDS-A

We start by analyzing how the uniform pattern delivers packets to individual networks and develop a simple model for the detection probability in IDS-A. We later extend this result to IDS-B.

Theorem 1: For $T \gg \Delta_s$, the probability that a normal subnet $s \in \mathcal{I}_N$ with IDS-A detects a uniform scanner is

$$\rho_A^s(r) \approx 1 - \Big(\sum_{j=0}^{a_s - 1} \binom{|s|}{j} q^j (1-q)^{|s|-j}\Big)^{1/q}, \tag{5}$$

where $q = \Delta_s/\omega T$.

Fig. 5 compares simulations to (5) as four of the main parameters of the model change. Numerical results indicate that (5) is accurate to within 1% as long as $T \ge 100\Delta_s$. For $T = 24$, this translates into $\Delta_s \le 14.4$ minutes. Part (b) shows one example where $T = 10\Delta_s$ is insufficiently large, which results in some mild discrepancy for values of $a_s \in [30, 35]$.

From the analysis of (5), observe that $\rho^s(r)$ is a function of product ωT, which means that increasing ω by a factor of m allows reduction of T by the same factor without changing

the detectability of the scanner. Thus, uniform scanning is IP-scalable against IDS-A *if and only if it uses post-permutation split*. For pre-permutation split (i.e., $\omega = 1$), the detection probability stays constant regardless of m and the scanner's stealthiness does not benefit from IP diversity.

D. Uniform Against IDS-B

For IDS-B, our first step is to understand inter-probe delays $\{Y_k^s\}_k$ seen by s from the attacker in our continuous model in Fig. 4. Note that the model allows some of these delays to span the border of multiple source IPs, which we deal with later in the section by requiring that $|s|/m$ be sufficiently large.

Theorem 2: Inter-probe delays $Y_1^s, \ldots, Y_{|s|-1}^s$ are identically distributed random variables with $E[Y_k^s] = \omega T/|s|$ and the following CDF tail

$$P(Y_k^s \geq y) = \left(1 - \frac{y}{\omega T}\right)^{|s|}, \quad 0 \leq y \leq \omega T. \tag{6}$$

Proof: First, notice that the uniform permutation is equivalent to randomly distributing $|s|$ points on a ring of length ωT. Since there are $|s|$ inter-probe gaps on the ring, their mean is simply $E[Y_k^s] = \omega T/|s|$. Second, the probability that a given address from s falls in the interval $[t, t+y) \subseteq [0, \omega T]$ is $y/\omega T$. Then, the probability that none of the addresses from s land into $[t, t+y)$ is $P(Y_k^s \geq y) = (1 - y/\omega T)^{|s|}$. ∎

We omit simulations showing that (6) is very accurate. Instead, we define $\chi_s = P(Y_k^s < \Delta_s)$ to be the probability that the uniform permutation sends two probes to s with spacing smaller than Δ_s and proceed to the next result.

Theorem 3: For $(|s|-a_s)(1-\chi_s)/m \to \infty$, the probability that IDS-B at a normal subnet $s \in \mathcal{I}_N$ detects a uniform scanner is asymptotically

$$\rho_B^s(r) \approx 1 - e^{-(|s|-a_s+1)(1-\chi_s)\chi_s^{a_s-1}}. \tag{7}$$

Proof: Define J_k^s to be an indicator variable of event $Y_k^s < \Delta_s$. Then, $P(J_k^s = 1) = 1 - P(J_k^s = 0) = \chi_s$. Since IDS-B needs $a_s - 1$ consecutive 1s in set $\{J_k^s\}_k$ to arrive into state a_s, define

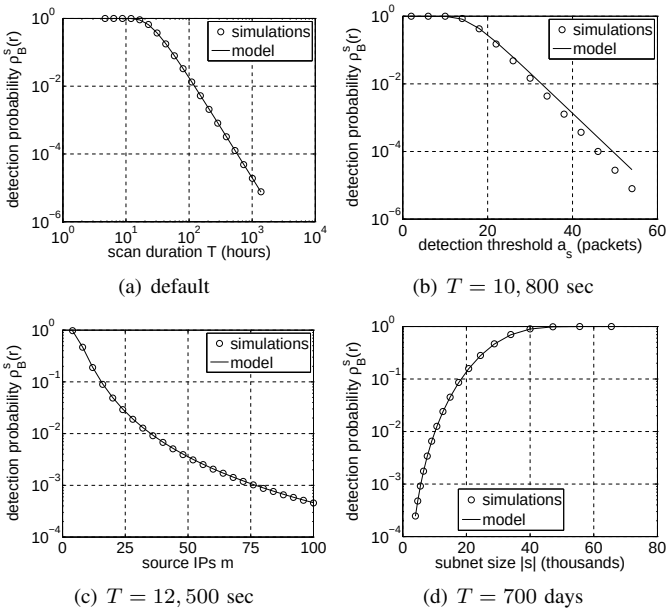
$$X_k^s = \begin{cases} 1 & J_k^s = J_{k+1}^s = \ldots = J_{k+a_s-2}^s = 1 \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

to be an indicator of a detection event occurring at time $k + a_s - 2$. Denoting by $l = |s| - a_s + 1$ the size of set $\{X_k^s\}_k$, we have that $X^s = \sum_{k=1}^{l} X_k^s$ is the total number of detections in $[0, \omega T]$ and $\rho^s(r) = P(X^s \geq 1)$.

Before deriving this probability, note that we need to analyze only those consecutive runs of 1s in sequence $\{J_k^s\}_k$ that follow a 0 and start no later than position l. Indeed, supposing that this set contains Z zeroes, X^s is non-zero if and only if any of the Z runs of 1s that immediately follow a zero has length of at least $a_s - 1$. All other runs provide redundant information and can be removed from consideration.

Define V_j to be the value of X_k^s following the j-th zero in set $\{J_k^s\}_{k=1}^{l}$. We then obtain

$$X^s = \sum_{j=1}^{Z} V_j. \tag{9}$$



(a) default (b) $T = 10,800$ sec

(c) $T = 12,500$ sec (d) $T = 700$ days

Fig. 6. Comparison of post-permutation IDS-B model (7) to simulations (default parameters $|s| = 2^8, \Delta_s = 60$ sec, $a_s = 4$, and $m = 1$).

From the Chen-Stein theorem [2] and treating set $\{J_k^s\}_k$ as approximately iid, variable X^s converges to the Poisson distribution with rate $\lambda = E[X^s] = E[Z]E[V_1^s]$ as $E[Z] \to \infty$. Noticing that $E[Z] = l(1-\chi_s)$ and $E[V_1^s] = \chi_s^{a_s-1}$, we get $\lambda = l(1-\chi_s)\chi_s^{a_s-1}$, which immediately leads to $\rho^s(r) \approx 1 - e^{-\lambda}$ in (7).

We should make two observations about this derivation. First, for small $|s|$ and large a_s, the dependency in set $\{J_k^s\}_k$ may be strong enough for $\rho^s(r)$ to disagree with the model (which arises because $\sum_k Y_k^s \leq \omega T$ and set $\{Y_k^s\}_k$ is not iid); however, in the limit (7) is exact. Second, although some delays Y_k^s may span between source IPs, condition $(|s| - a_s)(1-\chi_s)/m \to \infty$ ensures that *each* IP gets enough 0s in $\{J_k^s\}_k$ to invoke the Chen-Stein theorem and keeps the overall result asymptotically accurate. ∎

Fig. 6 compares simulations to (7) under the same default conditions as in Fig. 5. Results show that T, m, and $|s|$ do not influence the accuracy of the model if threshold a_s is small compared to $|s|$ (i.e., the error is below 0.1% for $a_s = 4$ and subnet sizes as small as 2^8). However, significantly larger a_s create too much dependency among consecutive delays $\{Y_k^s\}_k$ leading up to detection and result in a more serious mismatch with the model, as shown in part (b) of the figure. Despite this discrepancy, the model can be used to upper-bound $\rho_B^s(r)$ and compute scanning rates that guarantee a certain level of stealth.

As with IDS-A, uniform scanners are IP-scalable against IDS-B if and only if they use post-permutation split, which can be inferred from the ωT term in (6).

E. Uniform Cover Time

We next examine the time needed for the uniform permutation to cover a particular subnet. In order to determine this metric, we first relax the definition of SCT since uniform

scanners can never achieve $\rho^s(r) = 0$ with finite T. For a pattern X, define the ϵ-SCT $T_X^s(\epsilon)$ of a normal subnet $s \in \mathcal{I}_N$ to be the minimum duration T in which X can reduce the detection probability at s below ϵ, i.e., $T_X^s(\epsilon) = \inf\{t \geq 0 : \rho^s(n/t) \leq \epsilon\}$. We similarly relax the definition of k-stealthier and IP-scalable to operate in terms of ϵ-SCT instead of SCT.

This leads to the following approximation.

Theorem 4: Define $c = 1/(\beta - 1)$. Then, for $\epsilon \to 0$ and $|s| \gg \beta$, the ϵ-SCT of a β-aware uniform permutation is asymptotically

$$T_U^s(\epsilon) \approx \frac{\alpha|s|\Delta_s}{\omega} \begin{cases} e^{\eta_1}(\beta!)^{-c} & \text{IDS-A} \\ e^{\eta_2}\eta_3^{-1} & \text{IDS-B} \end{cases}, \tag{10}$$

where

$$\alpha = \Big(\frac{|s|}{-\log(1-\epsilon)}\Big)^c, \qquad \eta_1 = W(-c(\beta!)^c/\alpha), \tag{11}$$

$$\eta_2 = W(-c/\alpha), \qquad \eta_3 = \sum_{j=0}^{\infty} \frac{(\alpha e^{\eta_2})^{-j}}{j+1}, \tag{12}$$

and $W(.)$ is Lambert's function.

Proof: Since $\rho^s(r) = \epsilon$ is asymptotically small, one can make a number of approximations that greatly simplify inversion of (5) and (7). For small x, we use Taylor expansions $(1-x)^y \approx e^{-xy}$, $1 - e^{-x} \approx x$, and $\log(1-x) \approx -x$. We also neglect β in comparison to $|s|$, i.e., $|s| - \beta \approx |s|$.

Without a-priori knowledge of a_s, a uniform scanner must assume that counter $C_i^s(t)$ reaching β triggers detection for both IDS-A/B. This means (5) and (7) must undergo inversion with a_s replaced by β. For IDS-A and constant $|s|$, observe that $\epsilon \to 0$ implies $q \to 0$ and the leading term of ϕ_{bin}^s is

$$\phi_{bin}^s \approx \binom{|s|}{\beta} q^\beta (1-q)^{|s|} \approx \binom{|s|}{\beta} e^{\beta \log q - |s|q}. \tag{13}$$

Recalling that $\rho_A^s(r) \approx 1 - (1 - \phi_{bin}^s)^{1/q} = \epsilon$, we have

$$\log(1-\epsilon) \approx \frac{\log(1-\phi_{bin}^s)}{q} \approx \frac{-\phi_{bin}^s}{q}. \tag{14}$$

Using (13) in (14) and taking the log of both sides, we get

$$\log\Big(\frac{-\beta! \log(1-\epsilon)}{|s|^\beta}\Big) \approx (\beta - 1)\log q - |s|q. \tag{15}$$

This equation is of the general form $c = b\log q + aq$, whose solution using Lambert's $W(.)$ function is given by

$$q = \exp\Big[-W\Big(\frac{ae^{c/b}}{b}\Big) + \frac{c}{b}\Big]. \tag{16}$$

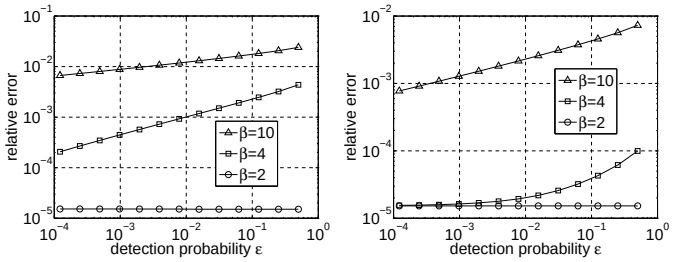
Applying this result to (15) and recalling that $q = \Delta_s/\omega T$, we arrive at the first line of (10).

For IDS-B, observe that (7) can be written as

$$-\log(1-\epsilon) \approx |s|(1-\chi_s)\chi_s^{\beta-1}. \tag{17}$$

Since $\chi_s \to 0$, we have $\log(1-\chi_s) \approx -\chi_s$ and

$$\log\Big(\frac{-\log(1-\epsilon)}{|s|}\Big) \approx -\chi_s + (\beta-1)\log\chi_s, \tag{18}$$



(a) IDS-A (b) IDS-B

Fig. 7. Relative error between the binary-search SCT and its closed-form approximations ($|s| = 2^{16}, \Delta_s = 60$ sec, $m = 1$).

which again has shape $c = a\chi_s + b\log\chi_s$. Solving (18) for χ_s, we get $\chi_s = e^{-\eta_2}/\alpha$, where η_2 and α are given in (11)-(12). Expanding $\chi_s = 1 - (1-q)^{|s|}$ and applying the log to both sides, we have

$$\frac{\log(1 - \frac{e^{-\eta_2}}{\alpha})}{|s|} \approx \log(1-q) \approx -q = \frac{-\Delta_s}{\omega T}. \tag{19}$$

Substituting $-\log(1-z) = z(1 + z/2 + z^2/3 + \ldots)$ with $z = e^{-\eta_2}/\alpha$ into (19), we get the second line of (10). ■

Fig. 7 shows the relative error between approximations (10) and the corresponding ϵ-SCT found using binary search on models (5), (7). For $\beta = 2$, the latter is so close to the former that their relative difference is initially less than 10^{-5}, which falls below Matlab's precision for binary search and explains why it does not improve as $\epsilon \to 0$. The other two curves in each subfigure show monotonic decay as ϵ decreases, with the IDS-B approximation generally better agreeing with the original than IDS-A. This arises from the extremely crude approximation to the binomial distribution in (13). For larger β, the error is generally more pronounced and decays slower since the magnitude of the omitted terms is higher; however, in all cases in the figure it stays below 2.4%.

IV. Optimal Scan Pattern

Our contribution in this section is to prove the existence of stealth-optimal scanners and analyze their model-based performance in comparison to uniform/sequential.

A. Local Pattern

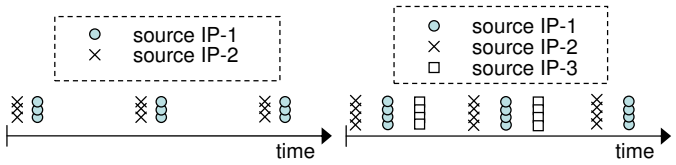
To understand optimal patterns, we next derive a lower bound on $\min_Y T_Y^s$ and show that there exists a *local* (i.e., as seen by each s) arrival pattern of packets that achieves optimality under both IDS-A/B. Later in the section, we develop a scanner that implements this pattern *globally* (i.e., simultaneously in all CIDR subnets).

Theorem 5: The SCT of $s \in \mathcal{I}_N$ is lower-bounded by

$$\min_Y T_Y^s \geq \frac{|s|\Delta_s}{m(\beta-1)}. \tag{20}$$

To show that STOP patterns exist, suppose each source i shapes its traffic to s into bursts of $\beta - 1$ packets separated by an intra-IP gap

$$\delta_{intra}^s = \frac{Tm(\beta-1)}{|s|}. \tag{21}$$



(a) STOP pattern ($\beta = 4, m = 2$) (b) STOP pattern ($\beta = 5, m = 3$)

Fig. 8. Stealthy β-aware probing seen by s.

As illustrated in Fig. 8, this pattern initially raises target count $C_i^s(t)$ to $\beta - 1$ and then follows it up with the proportionally-stretched gap in (21). Detection is avoided for IDS-B if and only if $\delta_{intra}^s > \Delta_s$. As discussed earlier, IDS-B is stricter than IDS-A, which means that the scanner also automatically avoids IDS-A. Combining the two cases and solving $\delta_{intra}^s > \Delta_s$ for T, this pattern exhibits the same SCT for both types of IDS

$$T_O^s = \frac{|s|\Delta_s}{m(\beta-1)}, \tag{22}$$

which is optimal as it equals the lower bound in (20).

Examining (22), notice that STOP patterns are not only IP-scalable, but also $(\beta-1)$-stealthier than any unaware pattern. Furthermore, the optimal SCT is a linear function of subnet size $|s|$ and all IDS parameters, unlike the uniform permutation whose SCT sometimes scales quadratically with $|s|$, which we establish later in this section during comparison analysis.

B. Global Pattern

In order to show the existence of global STOP scanners, we design a novel bursty schedule for GIW/RR, which is an Internet-wide scanning framework developed in [16] that relies on *Globally IP-Wide* (GIW) permutations and *Round-Robin* (RR) split. Due to limited space, our explanation of the relevant concepts from [16] is very brief.

Define $d = 32 - \lfloor \log_2(m(\beta-1)) \rfloor$ to be the number of left-most bits in the IP address (which we call the *depth*) at which subnets become size-trivial. In other words, subnets smaller or equal to $/d$ can never detect the scanner if it perfectly load-balances across its m source IPs. The main challenge in achieving STOP scanning is to ensure that target IPs arrive in bursts of $\beta - 1$ at subnets above depth d and are randomized according to GIW when viewed at subnets below d. Employing the *Alternating Gateway Tree* (AGT) of [16, Section 3.2], this can be accomplished by traversing the tree $\beta - 2$ times and flipping node directions *only* at depth no smaller than d. The last, i.e., $(\beta-1)$-st, traversal flips all 32 nodes to ensure that the next burst proceeds according to GIW.

Since AGTs are inefficient [16], the above algorithm must be implemented in practice using $\beta - 1$ *Reversed Linear Congruential Generators* (RLCGs) maintained by each source IP [16, Section 3.2]. Specifically, assume that the main RLCG is in position k in its GIW permutation $\{z_k\}_k$ and that the scanner needs to generate the next $\beta - 1$ targets $y_0, \ldots, y_{\beta-2}$ in a burst. The first target y_0 is simply z_k. Since the remaining $\beta - 2$ destinations do not change the top d levels of the tree, they can be found in the permutation where $\{z_k\}_k$ returns to the same subnet at level d. This is equivalent to skipping forward by 2^d elements each time. This leads to

$$y_j = z_{k+j2^d}, \quad j = 0, 1, \ldots, \beta - 2. \tag{23}$$

Algorithm 1 STOP at each scanner source IP

```
1: d = 32 − ⌊log2(m(β − 1))⌋                          ▷ Size-trivial depth
2: start = rand()                                      ▷ Initial seed in [0, 2^32 − 1]
3: totalB = 0                                          ▷ Total bursts generated
4: while start != EOS do                               ▷ While not end-of-sequence
5:     for j = 0 to β − 2 do
6:         LCG[j].Init(start)                          ▷ Set start value of LCG
7:         LCG[j].Skip(j2^d)                           ▷ Jump forward
8:     end for
9:     for k = 1 to 2^d do                             ▷ Iterate through 2^d bursts
10:        IP = totalB mod m                           ▷ Assigned source IP
11:        totalB++                                    ▷ Count burst number
12:        for j = 0 to β − 2 do
13:            x = LCG[j].Next()                       ▷ Get next LCG value
14:            if (x != EOS) ∧ (IP is ours) then
15:                y = ReverseBits(x)                  ▷ Obtain RLCG
16:                if y is valid then                  ▷ Falls in scanning space?
17:                    probe(y)                        ▷ Hit destination
18:                end if
19:            end if
20:        end for
21:        Sleep(T(β − 1)/n)                           ▷ Wait for next burst
22:    end for
23:    start = x                                       ▷ Next burst follows last hit target
24: end while
```

The entire process is summarized in Algorithm 1. After deciding the size-trivial depth and the starting IP in Lines $1-2$, the main loop in Line 4 runs through bursts of $\beta - 1$ packets until the random number generator (i.e., the LCG) wraps back to the original seed and returns a special EOS (end-of-sequence) IP address. To avoid re-generating the entire sequence for each burst, the scanner operates with $\beta-1$ LCGs, each pointing to a different part of the original sequence. Their initialization and advancement is shown in Lines $6-7$.

For each burst k, RR split decides in Line 10 which local IP will transmit the entire burst. Decisions about which targets to hit are made in Lines $13-14$ based on the current position of the underlying LCG and whether the burst is assigned to this particular source IP. Reversing the bits of the LCG in Line 15 and checking the result against the scanning space (e.g., IANA-allocated or BGP blocks) in Line 16 completes the main portion of the algorithm.

C. Comparison

We finish this section by analyzing the relative performance of the various algorithms. As $\epsilon \to 0$, the numerous constants in (10) disappear. Specifically, α becomes large and $\eta_1 \to 0, \eta_2 \to 0, \eta_3 \to 1$, which leads to

$$T_U^s(\epsilon) \approx \frac{|s|^{1+c}\Delta_s}{\omega\gamma\epsilon^c}, \quad \text{where } \gamma = \begin{cases} (\beta!)^c & \text{IDS-A} \\ 1 & \text{IDS-B} \end{cases}. \tag{24}$$

First, observe that uniform is stealthier against IDS-A by a factor of $(\beta!)^c$ than against IDS-B. This ratio is always no smaller than 2 and is approximately $(\beta/e)^{1+c}$ for $\beta \gg 1$.

While for sequential this ratio is always $\beta - 1$ and for STOP it is 1, the uniform permutation splits these two extremes one-third of the way (i.e., at $\beta/e \approx 0.37\beta$) as $\beta \to \infty$.

Second, notice that $T_U^s(\epsilon)$ is proportional to $|s|^{1+c}$, which may scale quite aggressively as $|s|$ becomes large (e.g., quadratically for $\beta = 2$). Because of this, sequential is actually stealthier than uniform for any s with $|s| > n_0$, where

$$n_0 = \left(\frac{n\gamma\epsilon^c}{\zeta}\right)^{\frac{\beta-1}{\beta}}, \tag{25}$$

which has not been previously documented and is quite counter-intuitive. For $\beta = 2$, this translates into $n_0 = \sqrt{\gamma n \epsilon}$. Assuming the desired detection probability $\epsilon = 10^{-3}$ (i.e., on average, one in $1,000$ subnets detects the scan), sequential is stealthier than uniform against IDS-A in any network with more than $2,930$ IPs and against IDS-B with more than $2,072$ IPs (i.e., these roughly map to $/20$ and $/21$, respectively). However, as β increases, (25) quickly rises as well. For $\beta = 4$, the corresponding thresholds are 14.1M (IDS-A) and 9.9M (IDS-B), which correspond to $/8$ or larger networks.

Third, even though for some scan patterns two sets of IDS-A parameters are equivalent if ratio $\Delta_s/(a_s - 1)$ (i.e., the average allowed gap between packets) remains constant, this is not the case against the uniform permutation. Lowering Δ_s while keeping the ratio constant actually *increases* the uniform cover time and makes IDS-A perform better at detecting the scanner. Thus, for example, combination $(15, 2)$ is much stricter against uniform scanners than Snort's default $(60, 5)$ although both allow on average 1 scan packet per 15-second interval.

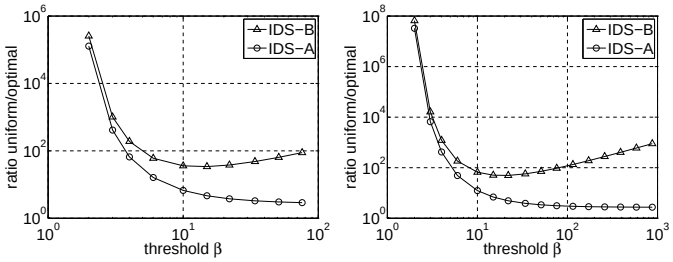
Fourth, comparing (22) with (2), notice that STOP is $n(\beta - 1)/\zeta|s|$ times stealthier than sequential in each s. Given a $/16$ subnet with Bro TRW's default $\beta = 4$, this translates into an improvement by a factor of 64K against IDS-A and 196K against IDS-B. This is equivalent to a reduction of T from 1 year to 8 minutes for IDS-A and 2.6 minutes for IDS-B, while keeping the detection probability the same. For a fixed T, STOP's inbound rate at each subset s is $\max(n/m|s|, 1)$ times smaller than sequential's. Using $T = 24$ hours and a modest $m = 10$, this results in 0.76 pps at $/16$ subnets and one probe every 337 seconds at $/24$ subnets, which is a reduction by a factor of 6.5K and 1.67M, respectively, over sequential.

Finally, notice from (22) and (24) that the stealth-optimal pattern is

$$\pi(\epsilon) = \frac{T_U^s(\epsilon)}{T_O^s} = \frac{|s|^c(\beta - 1)}{\gamma\epsilon^c} \tag{26}$$

times stealthier than uniform. This ratio is plotted in Fig. 9 for two subnet sizes. In both subfigures, (26) for IDS-A starts at $|s|/2\epsilon$ for $\beta = 2$ and converges toward e as $\beta \to \infty$. For IDS-B, it starts at double the IDS-A value and never drops below its global minimum $\pi_0 = e\log(|s|/\epsilon)$ achieved at $\beta_0 = \pi_0/e + 1$. This shows that regardless of β, the STOP pattern is at least π_0-stealthier against IDS-B than uniform. For the subfigures (a)-(b), these minimums are 34 and 49.

However, for small β the uniform pattern performs much worse, allowing $\pi(\epsilon)$ to reach 256K in Fig. 9(a) and 65M in Fig. 9(b). For $\beta = 4$ and the same $\epsilon = 10^{-3}$, STOP scanners in $/16$ subnets are 419-stealthier than uniform when facing IDS-A and 1,209-stealthier when facing IDS-B. This is equivalent to a reduction in T from 1 year to 21 hours in the former case and to 7 hours in the latter.



(a) $|s| = 2^8$ (b) $|s| = 2^{16}$

Fig. 9. Ratio $\pi(\epsilon)$ for $\epsilon = 10^{-3}$.

D. Discussion

One of the most peculiar results obtained in this paper is that all studied post-permutation scanners can linearly increase stealthiness with the number of source IPs. This leads to an intriguing idea of hijacking unused addresses from the same subnet when the infected device is not located behind a NAT[2]. To avoid possible detection from IP-address conflicts (which are reported to users and possibly administrators), worms can monitor ARP broadcasts and DHCP leases to silently drop IPs as soon as their legitimate owners join the network. In such scenarios, j stolen IPs by each infected host allow the scan to become not only j-stealthier (i.e., increase the speed by a factor of j for the same level of detection), but also much harder to map to the correct hardware without administrator access to ARP packets and MAC-layer addresses.

We finish the paper with practical implications. Assuming $T = 24$ hours and networks no larger than a $/16$, STOP can avoid the open-source version of Snort [35] using just $m = 12$ IPs, Bro [5] using 24 IPs, and Bro TRW [5] using 455 IPs, assuming their default settings. With a 12K-node botnet, where each host hijacks 10 local IPs using ARP, a stealth-optimal scanner can cover the Internet in one day and remain completely undetected in all $/8$ networks operating any Snort/Bro/TRW device with default parameters. This happens because none of the IDS are allowed to reach the threshold that trigger underlying detectors, which renders their accuracy irrelevant. While dropping threshold a_s or increasing interval Δ_s is possible, this may lead to unmanageably high rates of false-alarms [40], reduced administrator sensitivity, and lower operating efficiency of IDS.

Defenses against IP hijacking and methods for detecting distributed STOP scanners will be studied in future work.

V. Related Work

The first and most common direction for evading IDS involves sending malicious packets that do not match the

[2]NAT status of a host can be easily determined through public web servers such as `http://whatsmyip.org` and `http://ipchicken.com`.

signature database [13], [25], [30], [34]. Public tools such as nmap [25] rely on incorrect reconstruction of the packet at the IDS (e.g., using IP-level fragmentation [30], incorrect checksums, TTL tricks [34]), as well as the ability of the attacker to hide his/her identity and/or packet contents (e.g., using source-address spoofing, confusing IP options and flags [30], [34], and polymorphic packet contents [13]).

The second direction relies on concealing abnormal communication in ways that bypass IDS anomaly detectors [6], [39], [43]. Attackers can mimic benign traffic [6], [39] or modify scan rates [43] to avoid appearing like a propagating worm.

The third direction, which is the topic of this paper, works against pattern-based detectors by designing scan algorithms that do not allow IDS to reach its detection thresholds. We are aware of only one effort in this area, in which [12] alternates between known alive hosts in the target network and the remaining unexplored space to manipulate Bro TRW [10].

VI. Conclusion

This paper introduced a novel formalization of scanner algorithms and IDS detection rules related to horizontal scanning. We thoroughly investigated the detection probability of previous scan patterns and brought awareness to the existence of low-overhead algorithms for stealth-optimal scanning, which can remain undetected at much faster rates compared to the known approaches.

References

[1] M. Allman, V. Paxson, and J. Terrell, "A Brief History of Scanning," in *Proc. ACM IMC*, Oct. 2007, pp. 77–82.

[2] R. Arratia, L. Goldstein, and L. Gordon, "Two Moments Suffice for Poisson Approximations: The Chen-Stein Method," *The Annals of Probability*, vol. 17, no. 1, pp. 9–25, Jan. 1989.

[3] G. Bartlett, J. Heidemann, and C. Papadopoulos, "Understanding Passive and Active Service Discovery," in *Proc. ACM IMC*, Oct. 2007, pp. 57–70.

[4] D. Benoit and A. Trudel, "World's First Web Census," *Intl. Journal of Web Information Systems*, vol. 3, no. 4, pp. 378–389, 2007.

[5] Bro IDS. [Online]. Available: http://bro-ids.org/.

[6] P. Fogla, M. Sharif, R. Perdisci, O. Kolesnikov, and W. Lee, "Polymorphic Blending Attacks," in *Proc. USENIX Security*, Jul. 2006, pp. 241–256.

[7] Y. Gu, A. McCullum, and D. Towsley, "Detecting anomalies in network traffic using maximum entropy estimation," in *Proc. ACM/USENIX IMC*, Oct. 2005, pp. 345–350.

[8] J. Heidemann, Y. Pradkin, R. Govindan, C. Papadopoulos, G. Bartlett, and J. Bannister, "Census and Survey of the Visible Internet," in *Proc. ACM IMC*, Oct. 2008, pp. 169–182.

[9] B. Irwin and J. P. van Riel, "Using InetVis to Evaluate Snort and Bro Scan Detection on a Network Telescope," in *Proc. IEEE VizSEC*, Oct. 2007, pp. 255–273.

[10] J. Jung, V. Paxson, A. W. Berger, and H. Balakrishnan, "Fast Portscan Detection Using Sequential Hypothesis Testing," in *Proc. IEEE S&P*, May 2004, pp. 211–225.

[11] Juniper IDP. [Online]. Available: http://www.juniper.net/.

[12] M. G. Kang, J. Caballero, and D. Song, "Distributed Evasive Scan Techniques and Countermeasures," in *Proc. DIMVA*, Jul. 2007, pp. 157–174.

[13] O. M. Kolesnikov, D. Dagon, and W. Lee, "Advanced polymorphic worms: Evading IDS by blending in with normal traffic," Georgia Tech, Tech. Rep. GIT-CC-04-13, 2004.

[14] A. Lakhina, M. Crovella, and C. Diot, "Characterization of network-wide anomalies in traffic flows," in *Proc. ACM/USENIX IMC*, Oct. 2004, pp. 201–206.

[15] F. Lau, S. Rubin, M. Smith, and L. Trajkovic, "Distributed denial of service attacks," in *Proc. IEEE SMC*, Oct. 2000, pp. 2275–2280.

[16] D. Leonard and D. Loguinov, "Demystifying Service Discovery: Implementing an Internet-Wide Scanner," in *Proc. ACM IMC*, Nov. 2010, pp. 109–122.

[17] K. Levchenko, R. Paturi, and G. Varghese, "On the Difficulty of Scalably Detecting Network Attacks," in *Proc. ACM CCS*, Oct. 2004, pp. 12–20.

[18] Y. Li, Z. Chen, and C. Chen, "Understanding Divide-Conquer-Scanning Worms," in *Proc. IEEE IPCCC*, Dec. 2008, pp. 51–58.

[19] Z. Li, A. Goyal, Y. Chen, and V. Paxson, "Automating Analysis of Large-Scale Botnet Probing Events," in *Proc. ACM ASIACCS*, Mar. 2009, pp. 11–22.

[20] P. K. Manna, S. Chen, and S. Ranka, "Exact Modeling of Propagation for Permutation-Scanning Worms," in *Proc. IEEE INFOCOM*, Apr. 2008, pp. 1696–1704.

[21] D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, and N. Weaver, "Inside the Slammer Worm," *IEEE Security and Privacy Magazine*, vol. 1, no. 4, pp. 33–39, 2003.

[22] D. Moore, C. Shannon, D. J. Brown, G. M. Voelker, and S. Savage, "Inferring Internet denial-of-service activity," *ACM TOCS*, vol. 24, no. 2, pp. 115–139, 2006.

[23] D. Moore, C. Shannon, and J. Brown, "Code-Red: A Case Study on the Spread and Victims of an Internet Worm," in *Proc. ACM IMW*, Nov. 2002, pp. 273–284.

[24] NetVCR. [Online]. Available: http://www.niksun.com/.

[25] Nmap. [Online]. Available: http://nmap.org/.

[26] R. Pang, V. Yegneswaran, P. Barford, V. Paxson, and L. Peterson, "Characteristics of Internet Background Radiation," in *Proc. ACM IMC*, Oct. 2004, pp. 27–40.

[27] N. Provos and P. Honeyman, "ScanSSH - Scanning the Internet for SSH Servers," in *Proc. USENIX LISA*, Dec. 2001, pp. 25–30.

[28] Y. Pryadkin, R. Lindell, J. Bannister, and R. Govindan, "An Empirical Evaluation of IP Address Space Occupancy," USC/ISI, Tech. Rep. ISI-TR-2004-598, Nov. 2004.

[29] Psad: Intrusion Detection and Log Analysis with iptables. [Online]. Available: http://www.cipherdyne.org/psad/.

[30] T. H. Ptacek and T. N. Newsham, "Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection," Secure Networks, Inc., Tech. Rep., Jan. 1998.

[31] S. Resnick, *Adventures in Stochastic Processes*. Birkhäuser, 2002.

[32] S. Schechter, J. Jung, and A. W. Berger, "Fast Detection of Scanning Worm Infections," in *Proc. RAID*, Sep. 2004, pp. 59–81.

[33] C. Shannon and D. Moore, "The Spread of the Witty Worm," *IEEE Security and Privacy Magazine*, vol. 2, no. 4, pp. 46–50, 2004.

[34] S. Siddharth, "Evading NIDS, revisited," Dec. 2005. [Online]. Available: http://www.securityfocus.com/infocus/1852.

[35] Snort IDS. [Online]. Available: http://www.snort.org/.

[36] A. Soule, K. Salamatian, and N. Taft, "Combining Filtering and Statistical Methods for Anomaly Detection," in *Proc. ACM/USENIX IMC*, Oct. 2005, pp. 331–344.

[37] S. Staniford, J. A. Hoagland, and J. M. McAlerney, "Practical Automated Detection of Stealthy Portscans," *Computer Security*, vol. 10, no. 1–2, pp. 105–136, 2002.

[38] S. Staniford, V. Paxson, and N. Weaver, "How to 0wn the Internet in Your Spare Time," in *Proc. USENIX Security*, Aug. 2002.

[39] K. Tan, K. Killourhy, and R. Maxion, "Undermining an Anomaly-Based Intrusion Detection System Using Common Exploits," in *Proc. RAID*, 2002, pp. 54–73.

[40] G. C. Tjhai, M. Papadaki, S. Furnell, and N. L. Clarke, "Investigating the problem of IDS false alarms: An experimental study using Snort," in *Proc. IFIP SEC*, Sep. 2008, pp. 253–267.

[41] N. Weaver, S. Staniford, and V. Paxson, "Very Fast Containment of Scanning Worms," in *Proc. USENIX Security*, Aug. 2004, pp. 29–44.

[42] J. Xia, S. Vangala, J. Wu, L. Gao, and K. Kwiat, "Effective Worm Detection for Various Scan Techniques," *Computer Security*, vol. 14, no. 4, pp. 359–387, Jul. 2006.

[43] W. Yu, X. Wang, P. Calyam, D. Xuan, and W. Zhao, "On Detecting Camouflaging Worm," in *Proc. ACSAC*, Dec. 2006, pp. 235–244.

[44] C. C. Zou, D. Towsley, and W. Gong, "On the Performance of Internet Worm Scanning Strategies," *Elsevier Performance Evaluation*, vol. 63, no. 7, pp. 700–723, Jul. 2006.