Analysis of Network Address Shuffling as a Moving Target Defense

Thomas E. Carroll
Pacific Northwest National Laboratory
Richland, Washington 99352
Email: Thomas.Carroll@pnl.gov

Michael Crouse
Computer Science Dept.
Harvard University
Cambridge, Massachusetts 02138
Email: mcrouse@seas.harvard.edu

Errin W. Fulp and Kenneth S. Berenhaut
Computer Science and Mathematics Depts.
Wake Forest University
Winston-Salem, North Carolina 27109
Email: {fulp, berenhks}@wfu.edu

***Abstract*—Address shuffling is a type of moving target defense that prevents an attacker from reliably contacting a system by periodically remapping network addresses. Although limited testing has demonstrated it to be effective, little research has been conducted to examine the theoretical limits of address shuffling. As a result, it is difficult to understand how effective shuffling is and under what circumstances it is a viable moving target defense.**

This paper introduces probabilistic models that can provide insight into the performance of address shuffling. These models quantify the probability of attacker success in terms of network size, quantity of addresses scanned, quantity of vulnerable systems, and the frequency of shuffling. Theoretical analysis shows that shuffling is an acceptable defense if there is a small population of vulnerable systems within a large network address space, however shuffling has a cost for legitimate users. These results will also be shown empirically using simulation and actual traffic traces.

I. Introduction

An important phase of many cyber attacks is system reconnaissance, where the attacker seeks to gather information about the potential victim. Discovering the victim's IP address, operating system, and network services, is important since it can be used to determine an appropriate exploit mechanism. It has been reported that an attacker can be expected to spend upwards of 45 percent of their time performing reconnaissance [1]. Therefore given the amount of resources and the value associated with the information gathered, mitigating the reconnaissance attack phase can be an effective defense strategy.

Moving Target (MT) defenses are a type of diversity defense that constantly alters the attack surface of a target. In the time it takes the attacker to perform reconnaissance the MT defense will have changed the targeted system in such a way as to render the attacker's knowledge ineffective or stale. As a result the attacker will act on false information that can result in expending more resources and increasing the risk of detection.

While there are several different types of MT defenses [2], this paper investigates the performance of network address shuffling. Network address shuffling is a MT defense that dynamically alters an organization's network by remapping the usually static association between addresses and systems. Even though networks naturally experience address churn [3], the periods between address reassignments are ample enough to permit attackers to gather intelligence over a period of time and then exploit the information at a later date. Address shuffling methodically shortens the intervening periods, rendering intelligence stale more quickly.

Several different network address shuffling implementations exist [2]–[5]; however, the performance evaluation has largely been empirical. As a result, the findings of these studies are typically limited to very specific scenarios (e.g. certain network address space size). Theoretical models can provide a better understanding of the performance of shuffling under various conditions. For example theoretical models have been used to investigate the performance of Address Space Layout Randomization (ASLR) [6]. Using ASLR as a MT defense, computer memory is dynamically remapped to prevent an attacker from reliably discovering the exact layout of a targeted program in memory. ASLR does have similarities to network address shuffling (dynamically moving a targeted item around a fixed space), therefore some of the findings in this paper will be consistent with [6]. For example randomly moving targets around a space has little benefit if the amount of computer memory or network address space is small.

This paper develops theoretical models to describe the defense benefits of network address shuffling under various conditions. *Urn models* are developed to examine attacker success under static addresses and perfect shuffling. A network is not permuted with *static addresses*. The optimal strategy of the attacker is to explore the entire address space, never contacting the same host twice. On the other hand, *perfect shuffling* results in the network addresses being updated after each and every connection. Connecting to the same address results in different hosts being contacted. Even if an attacker sequentially probes the entire address space, it is unlikely that all hosts would be discovered. Models provide the necessary tools to demonstrate the effects that network size, quantity of vulnerable systems, quantity of addresses scanned, and the frequency of shuffling have on the *probability* attacker success.

Contribution of this paper. This paper introduces a set of probabilistic models to investigate the performance of address shuffling in defending networks. The models quantify the attacker success in terms of network size, number of addresses explored, and number of vulnerable systems. Using simulation and actual network traces, the relationship between shuffling frequency and connection loss is also investigated. Results will show that shuffling provides limited protection against attackers seeking just one high-value system. These results are also verified via real network traces replayed as empirical verification. This work is novel as related work performs a strategic analysis of defending honeypots with network shuffling.

Organization. This paper is organized as follows. In Section II we discuss the motivation for network address shuffling and describe current implementations of the method. In Section III we develop probabilistic models for network shuffling; this is then used in Section IV to analyze network defenses that incorporate shuffling. Section V summarizes the performance of address shuffling as a MT defense and introduces some potential future work.

II. Network Attacks and Shuffling Techniques

As mentioned in Section I, intelligence gathered during reconnaissance is extremely valuable to the attacker since it is used to prepare and develop subsequent attack plans. Network reconnaissance actions include the enumeration of the address space to discover key targets, such as databases and email servers. Tools such as Nmap (http://nmap.org/), a network scanner, and Shodan (http://www.shodan.com/), a computer services search engine, are commonly used to scan and enumerate a target network. Degrading the performance of these network reconnaissance tools using a MT defense within the organization's boundaries has been shown to be practical. The additional uncertainty causes the attacker to either expend additional efforts in casing and attack preparations or assume greater risk in the subsequent attack.

A. Reconnaissance Defenses and Shuffling Techniques

A common reconnaissance defense relies on network firewalls to prevent the attacker from discovering devices—or more specifically services. Although this approach is somewhat useful, the identity of devices that require a constant and public external presence (i.e., web and email servers) are not protected. Therefore, a more dynamic defense can provide improved security.

Dynamic network reconfiguration is an attempt to create unpredictable change, thereby creating a MT defense. After the attacker has performed reconnaissance, reconfiguration is employed to devalue the knowledge. As one might expect, the temporal aspect of dynamic network reconfiguration plays a critical role. Examples of dynamic reconfiguration includes unpredictable server selection and unpredictable route selection [4], [7]; however, this paper focuses on network address shuffling.

Network address shuffling is a dynamic reconnaissance defense that periodically permutes the relationships between addresses and devices. For the Internet, addresses are a combination of IP and transport layer information (protocol and port numbers) and either or both types of information can be used for shuffling; however without loss of generality this paper will only consider IP addresses. IP address shuffling replaces the address of protected devices within a network with a pseudo-random IP address chosen from the address space available to the administrator.

Two important components of any method are how addresses are mapped and how legitimate hosts stay in contact after a shuffle event. For example, Network Address Space Randomization (NASR) shuffles addresses using Dynamic Host Configuration Protocol (DHCP) to reassign addresses and relies upon Domain Name Service (DNS) to reestablish connectivity after a shuffle event [3]. Using this approach, timers are used to notify devices when their current address leases have expired. Once addresses have been reassigned, DNS entries are updated with the new information and applications that know the hostname of a device can query for the updated address. Of course this approach provides no defense for attackers that also know the hostname of their targets, such as a hit-list worm [3].

Dynamic Network Address Translation (DYNAT) [1] and Applications that Participate in their Own Defense (APOD) [4] are similar shuffling approaches that rely upon a form of Network Address Translation (NAT) for reassigning addresses. In these techniques, the NATed addresses provided to the external network to connect to internal devices can be periodically remapped. Hosts that are legitimately communicating are able to maintain connection by knowing the key associated with the algorithm that performs the shuffling. This approach does not rely on DNS and therefore is immune to threats that incorporate target lists, but it does require communicating hosts to share *a priori* knowledge about the shuffling algorithm and parameters. A further drawback of this approach is that existing threats already present in the internal network are minimally impacted.

Although different in implementation, limited testing has demonstrated that address shuffling methods are effective reconnaissance defenses. The defense performance (probability of attacker successfully finding a target) is a function of several parameters, including the size of the address pool, number of devices in the network to protect, the number of devices required to be successfully found by the attacker, and the shuffle frequency. In many cases these parameters have been studied empirically; this paper provides probabilistic models to study the effectiveness of shuffling.

III. Models for Network Shuffling Defenses

Consider the scenario in which an attacker targets a network defended by an administrator. As we described in the previous section, the attacker seeks to perform reconnaissance on the network while the administrator wants to keep hidden the identity (IP addresses) of vulnerable computers. As such, it is possible to model the system to determine if and when address shuffling is advantageous. Assume the following is true with regards to the attackers capabilities, the administrator defenses, and the network.

- There are n total addresses available to the administrator (address space) and $v \leq n$ vulnerable computers.

- A shuffle event randomly and uniformly remaps all n addresses in the network.
- The attacker is aware of the address space (n addresses) and will serially attempt k connections (probes).
- The goal of the attacker is to contact at least one of the v vulnerable computers in k attempts.

Two statistics are of interest 1) attacker success probability and 2) expected number of probes required to find a vulnerable computer. While it may be possible to model this system in several different fashions, this paper uses urn-based models to determine these important performance statistics. Note, a more detailed model description can be found in [8].

A. Urn-Based Models

A simple urn model consists of a vessel containing a set of marbles of different colors. The player randomly draws a marble from the urn and records its color. Although conceptually simple, urn models have been successfully used to model outcomes of various complex systems in physics, communication theory, and computer science [9].

For address shuffling, consider an urn containing v green and $n - v$ blue marbles for a total of n marbles. The green marbles represent vulnerable systems, while the blue marbles represent non-vulnerable systems. A non-vulnerable system could be a non-vulnerable computer (for example, a system that is patched or is not providing the exploitable service) or simply an address that is not associated with a device. This population of marbles represents the network address space of the administrator.

The player (attacker) draws, one at a time, k marbles from the urn. If the attacker draws at least one green marble (again, green represents vulnerable systems) then the set of draws is considered a success. Altering what happens between draws can be used to represent different defense strategies. Let us first consider the extremes of shuffling: static addressing (no shuffling) and perfect shuffling (shuffle after each probe).

B. Modeling Static Addresses

Assume the addresses assigned to computers in the network are fixed. Given this situation, the attacker's strategy for finding a vulnerable system is to sequentially iterate through the address space. If $k \geq n$ then the attacker is assured to find all v vulnerable computers. However, if $k < n$ then an urn model can be used to determine the likelihood of attacker success.

Consider the urn model where a drawn marble is never replaced. The attacker success probability then is determined via a hypergeometric distribution which determines the *"number of successes in a sequence of k draws from a finite population without replacement"* [9]. If X_k is a random variable representing the number of green marbles drawn (vulnerable computers contacted) given k draws then

$$\Pr(X_k = x) = \frac{\binom{v}{x}\binom{n-v}{k-x}}{\binom{n}{k}}. \quad (1)$$

As applied to computer networks, the *no replacement* requirement models the knowledge gained per probe since once an IP address has been probed there is no need to contact it again. The probability that at least one vulnerable system is discovered is then

$$\Pr(X_k > 0) = 1 - \Pr(X_k = 0) = 1 - \frac{\binom{n-v}{k}}{\binom{n}{k}}. \quad (2)$$

The number of probes needed to find a vulnerable computer can be modeled as a negative hypergeometric distribution, which describes the number of draws, without replacement, required to obtain a specific population of marbles [9]. For static addresses, if Y is a random variable representing the number of probes then the expected number of probes is

$$E[Y] = \frac{n+1}{v+1}. \quad (3)$$

C. Modeling Perfect Address Shuffling

Address shuffling remaps network addresses to the devices connected to a network. As previously stated, assume shuffling remaps all n addresses in the network, therefore at the end of a shuffling event the attacker loses any reconnaissance knowledge gained.

As described in [3], the frequency of shuffling events will impact the success rate of the attacker. One extreme case is *perfect shuffling*, where the administrator shuffles after every reconnaissance attempt. The attacker gains limited knowledge from one reconnaissance attempt to the next. If the attacker is allowed $k = n$ attempts, it is unlikely that all systems are contacted and the resulting probability of success rate is less than one.

Consider a model in which drawn marbles are returned to the urn. Given this requirement the attacker success probability is determined via a binomial distribution which determines the *"number of successes in a sequence of k draws from a finite population with replacement."* If X_k is a random variable representing the number of green marbles given k attempts then

$$\Pr(X_k = x) = \binom{v}{x} p^x (1-p)^{k-x} \quad (4)$$

where $p = \frac{v}{n}$ which is the probability of drawing a green marble (vulnerable computer), and $k \geq x$. Therefore, the probability of attacker success given k probes is

$$\Pr(X_k > 0) = 1 - Pr(X_k = 0) = 1 - (1-p)^k. \quad (5)$$

The expected number of probes needed to find a vulnerable computer can be modeled by a geometric distribution, which describes the number of draws with replacement required before a marble of a given color is drawn [9]. Assuming a perfect shuffling defense, if Y is a random variable representing the number of probes then it is easy to show the expected number of probes is

$$E[Y] = \frac{1}{p} = \frac{n}{v}. \quad (6)$$

The next section will use these models to analyze the effectiveness of network address shuffling.

IV. Analysis of Shuffling Defenses

The previous section developed performance equations for static addresses and perfect shuffling, the two extreme cases for network address shuffling. These performance equations are dependent on several variables, such as the number of probes, number of vulnerable computers, and shuffling frequency. This section will analyze the impact of these variables on the benefit and cost of static addressing and perfect shuffling under certain conditions. Although not exhaustive, the examples will provide guidance in the use of shuffling. In addition, the benefit and cost of shuffling less frequently will be studied empirically.

A. Network Address Space (one vulnerable computer)

Assume there is only one vulnerable computer in the network and the attacker is able to probe the entire network space (number of probe attempts equals the number of addresses in the space). In this case, static addresses provide no defense since the attacker can simply iterate through the address space until the vulnerable computer is discovered. In contrast if perfect shuffling is used then the success probability of the attacker will depend on network size.

Figure 1 shows the success probability as the network size increases where the attacker is permitted to probe the entire network space. As seen in the graph, the attacker is likely to find the vulnerable computer when the network size is small; however, the success rate drops as the network size increases and converges to $e^{-1} \approx 0.63$ which is the limit of (5) when $k = n$,

$$\Pr(0 < X_n \leq n) = 1 - \left(1 - \frac{1}{n}\right)^n. \tag{7}$$

In this case perfect shuffling reduces the probability of attacker success by 37% as compared to using static addresses.

B. Percentage of Address Space Probed

Another important parameter that impacts attacker success is the number of probes (scans) allowed. Again assume only one vulnerable computer exists in the network. In this case the probability of success given static addresses increases linearly as number of probes increases, since the probability that k probes will find the vulnerable computer is k/n. For example, if half of the network can be probed then the probability of success would be 0.5. For perfect shuffling, the success probability defined by (5) becomes

$$\Pr(0 < X_n \leq k) = 1 - \left(1 - \frac{1}{n}\right)^k. \tag{8}$$

As we increase the number of allowed probes, k, the success probability slowly increases. As described in the preceding subsection, the maximum success probability is approximately 0.63 when the entire network can be probed. This is also shown in Fig. 2, which depicts the success probability for the attacker as the number of probes increases, given a single vulnerable computer in a class-C network (255 addresses). Comparing perfect shuffling to static addressing, perfect shuffling provides an improved defense.

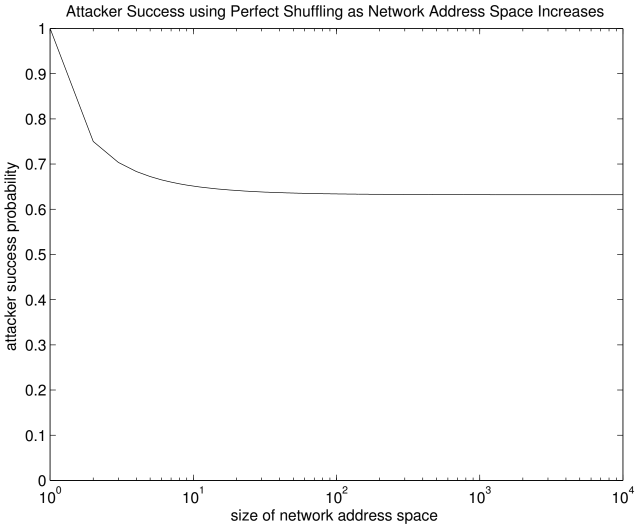

Fig. 1. The attacker success probability for finding the vulnerable computer as the network size increases. The attacker can make n probes, which is the number of addresses in the network. Note, the the x-axis is in log scale.

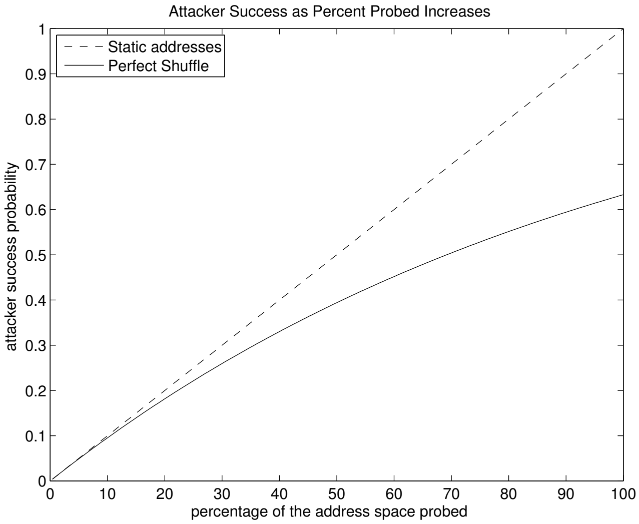

Fig. 2. The attacker success probability for finding the vulnerable computer as the number of network probes increases. Only one vulnerable computer exists in the class-C network space.

C. Number of Vulnerable Computers

The number of vulnerable computers is another key parameter for the success of the attacker. The previous examples assumed only one vulnerable computer exists in the network. Now consider the scenario that the attacker must discover one vulnerable computer in a network containing multiple targets. Clearly as the number of vulnerable computers increases the likelihood of finding one will increase. Assuming the attacker performs $k = n$ probes, then static addresses provides no

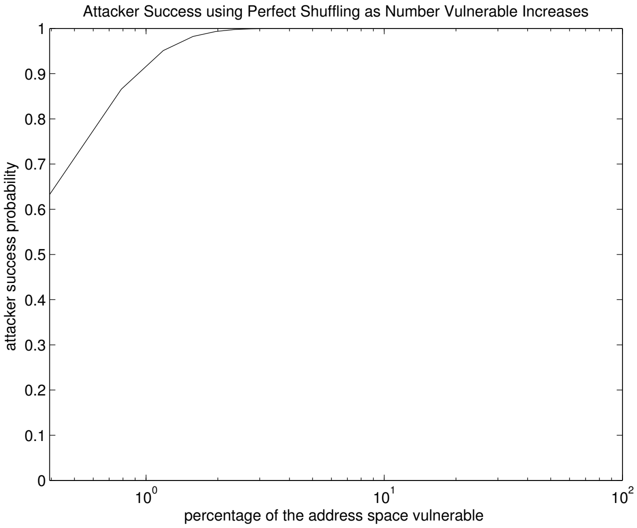

Fig. 3. The attacker success probability for contacting one vulnerable computer as the total number of vulnerable computers increases. The attacker performs $k = n$ probes, where n is the number of addresses in the network.

defense. If perfect shuffling is used then the success probability given by (5) reduces to

$$\Pr(0 < X_n \leq k) = 1 - \left(1 - \frac{v}{n}\right)^n. \tag{9}$$

Therefore the success probability approaches one as the number of vulnerable computers increases. Again, the goal is to locate at least one vulnerable computer out of v. This is shown in Fig. 3, which depicts the attacker success probability for a class-C network where the attacker performs n probes (number of addresses in the network). Therefore perfect shuffling is only beneficial if there is a relatively small number, less than 1% in this example, of vulnerable computers in the network.

D. Expected Number of Probes

The expected number of probes required for attacker success given static addresses and perfect shuffling were defined by equations (3) and (6) respectively. Note a defense that requires more probes by the attacker is considered better.

Comparing these equations, perfect shuffling always requires more probes; however, this advantage decreases as the percentage of systems that are vulnerable increases. A similar finding was described in Section IV-C, where perfect shuffling provided little advantage if the attack was permitted to probe the entire network space (number of probes equals the number of addresses in the space) and a significant percentage of computers where vulnerable.

E. Address Shuffle Frequency

Section III derived probabilistic models to estimate the performance of static addressing and perfect address shuffling. However, as previously mentioned, the models do not consider the performance of shuffling less frequently, for example, shuffling after every q probes. In this case the cost of shuffling with regards to lost legitimate connections must also be considered. For example, many shuffling techniques will cause active connections to immediately drop after a shuffle event.

In the case of NASR, a connection can only be restored after the new DNS entries have been distributed [3]. Therefore one shuffling cost to consider is the probability that a legitimate connection will be dropped. Studying this shuffling cost, referred to as the drop probability, is problematic since it relies on assumptions about network connection arrival patterns. Here, as in [3], the performance and cost of shuffling are studied empirically in this section.

Network traces were collected from the Dartmouth University CRAWDAD project which contains traffic occurring across a campus [10]. For each trace, the TCP connection start times and durations were identified. A network simulation program was then developed to model a class-C IP network defended using address shuffling. The connection information was used to model arriving connections, maintaining the arrival times and durations, however, the simulator randomly associated the destination of each connection with an address within the class-C address space. The attacker is allowed to make 255 probes, in an attempt to find one of 10 vulnerable systems within the network. The probes occur serially, each lasting 32 milliseconds, which is commensurate with the model developed in the previous section.

The shuffling rate varied from never occurring (static addresses) to shuffling after every probe (perfect shuffling). The shuffling rate will be described as a normalized value, where zero represents no shuffling (static addresses) and one is perfect shuffling. Therefore a shuffle rate of 0.5 indicates that a shuffle event occurred after half of the probes were attempted (again for this example, the attacker was allowed 255 probes). For each possible shuffle rate 100,000 experiments were performed. The average connection loss, attacker success probabilities, and expected number of vulnerable computers contacted were recorded.

Figures 4 show the result of the experiments, where the shuffle rate ranges from static addressing to perfect shuffling. At zero shuffle rate (static addressing), no connections are lost; however, the attacker has a 100 percent chance of finding a vulnerable system. As the shuffling rate increases, the attacker success probability drops which is predicted by the theoretical models in the previous sections. When perfect shuffling is employed the success probability is approximately 0.63 percent, again predicted by the theoretical models. However, as the shuffle rate increases the connection loss probability also increases. The loss probability increases slowly until reaching approximately 0.85 shuffle rate (corresponds to shuffling every 38 probes). For this system where the attacker must find one of 10 vulnerable systems within the network, a lower shuffling rate can provide a comparable defense to perfect shuffling but with a lower loss probability.

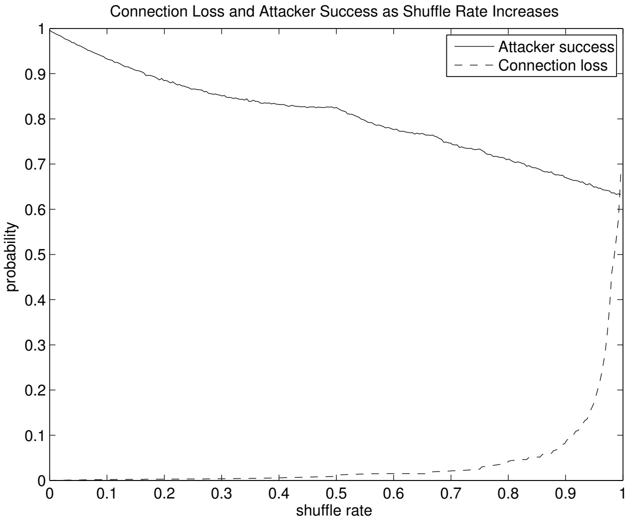

Fig. 4. The average percentage of vulnerable computers contacted as the normalized shuffle rate increases. A shuffle rate zero is static addressing, while a shuffle rate of 1 is perfect shuffling (shuffle after each attacker probe). Network contained 10 vulnerable computers in a class-C address space.

V. Conclusions

Moving Target (MT) defenses seek to render this information useless by constantly changing the targeted system. While there have been several studies into MT defenses, many results are empirical in nature. This paper introduces a set of urn-based models for theoretically measuring the performance of network shuffling, a MT method that periodically remaps the relations between network addresses and systems within a network.

The performance of static addresses (no shuffling) and perfect shuffling (shuffle after every connection) is dependent on several variables including the network size, number of vulnerable computers, and the amount of the address probed by the attacker. Given these variables, the developed equations can serve as valuable tools to determine if and when address shuffling provides a security benefit. For the attack scenario considered in this paper (attacker seeks to contact at least one vulnerable computer), analysis indicates shuffling provides some protection for networks that have very few vulnerable systems; otherwise shuffling provides limited benefit. In addition the expense of shuffling (impact on legitimate connections) might be considered too high for realistic use.

The models demonstrate that the protection offered by shuffling is a function, among other parameters, of address pool size. Larger pools present more opportunities to hide systems. However generally speaking, the models and attack scenarios presented in this paper indicate shuffling has defense benefit that is likely less than originally perceived. Furthermore, we must not forget that networks exist to serve users. A network is of limited benefit if users cannot find and reliably employ services. When observed from the network level, shuffling creates the appearance of a network in disarray. But at the user level, the network is calm and motionless. The network would also appear motionless to an attacker who has gained a user level perspective. In practice, an attacker accomplishes this by discovering the DNS name of a system, querying service location databases, or compromising workstations that have been keyed for the shuffle mechanism (via, for example, drive-by download of malware). Once this occurs—and no doubt it will occur as the adversaries evolve their tactics in response to moving target defenses—the size of the pool doesn't matter.

This work serves as an important first step in understanding the performance of address shuffling as a dynamic defense technique, and there are other important issues that can be further developed and analyzed. For example, a more comprehensive study of the impact of the different variables associated with shuffling can be explored along with the use of network shuffling in combination with other defenses (e.g., honeypots, network telescopes). This paper analyzes the extreme cases of shuffling (zero shuffling to perfect shuffling). The development of a theoretical model for infrequent shuffling would be beneficial and is currently under consideration. Beyond infrequent shuffling, the model can be employed to study parallel probes, in which the attacker performs multiple probes between shuffle events.

Acknowledgment

Portions of the research were funded by PNNL's Assymmetric Resilient Cybersecurity & Development Initiative. The views expressed in this paper are the opinions of the authors, and do not represent the official positions of the Pacific Northwest National Laboratory or the US Department of Energy.

References

[1] D. Kewley, R. Fink, J. Lowry, and M. Dean, "Dynamic approaches to thwart adversary intelligence gathering," in *Proc. of the DARPA Information Survivability Conference & Exposition II (DISCEX '01)*, vol. 1, 2001, pp. 176–185.

[2] M. Dunlop, S. Groat, R. Marchany, and J. Tront, "Implementing an IPv6 moving target defense on a live network," in *Proc. of the National Symposium on Moving Target Research*, 2012.

[3] S. Antonatos, P. Akritidis, E. P. Markatos, and K. G. Anagostakis, "Defending against hitlist worms using network address space randomization," *Computer Networks*, vol. 51, pp. 3471–3490, 2007.

[4] M. Atighetchi, P. Pal, F. Webber, and C. Jones, "Adaptive use of network-centrick mechanisms in cyber-defense," in *Proc. of the 6th IEEE Int. Symp. On Object-Oriented Real-Time Distributed Computing (ISORC '03)*, 2003, pp. 183–192.

[5] P. Beraud, A. Cruz, S. Hassell, and S. Meadows, "Using cyber maneuver to improve network resiliency," in *Proc. of the 2011 IEEE Military Communications Conference (MILCOM 2011)*, 2011, pp. 1121–1126.

[6] D. Evans, A. Nguyen-Tuong, and J. Knight, "Effectiveness of moving target defenses," in *Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats*, S. Jajodia, A. K. Ghosh, V. Swarup, C. Wang, and X. S. Wang, Eds. Springer, 2011, pp. 29–48.

[7] J. Michalski, C. Price, E. Stanton, E. Lee, K. S. Chua, Y. H. Wong, and C. P. Tan, "Final report for the network security mechanisms utilizing network address translation LDRD project," Sandia National Laboratory, SAND Rep. SAND2002-3613, Nov. 2002.

[8] M. Crouse, "Performance analysis of cyber deception using probabilistic models," Master's thesis, Wake Forest University, 2012.

[9] H. M. Mahmoud, *Pólya Urn Models*. Chapman and Hall, 2008.

[10] D. Kotz, T. Henderson, and I. Abyzov, "CRAWDAD trace dartmouth/campus/tcpdump/fall03 (v. 2004-11-09)," Nov. 2004, accessed on Nov 11, 2010 from: http://crawdad.cs.dartmouth.edu/dartmouth/campus/tcpdump/fall03.