Probabilistic Performance Analysis of Moving Target and Deception Reconnaissance Defenses

Michael Crouse
School of Engineering and Applied Sciences
Harvard University
Cambridge, MA, USA
mcrouse@seas.harvard.edu

Bryan Prosser
Department of Computer Science
Wake Forest University
Winston-Salem, NC, USA
prosbj9@wfu.edu

Errin W. Fulp
Department of Computer Science
Wake Forest University
Winston-Salem, NC, USA
fulp@wfu.edu

ABSTRACT

Deception and moving target reconnaissance defenses are techniques that attempt to invalidate information an attacker attempts to gather. Deception defenses attempt to mislead attackers performing network reconnaissance, while moving target defenses seek to make it more difficult for the attacker to predict the state of their target by dynamically altering what the attacker sees. Although the deployment of reconnaissance defenses can be effective, there are nontrivial administration costs associated with their configuration and maintenance. As a result, understanding under the circumstances these defenses are effective and efficient is important.

This paper introduces probabilistic models for reconnaissance defenses to provide deeper understanding of the theoretical effect these strategies and their parameters have for cyber defense. The models quantify the success of attackers under various conditions, such as network size, deployment of size, and number of vulnerable computers. This paper provides a probabilistic interpretation for the performance of honeypots, for deception, and network address shuffling, for moving target, and their effect in concert. The models indicate that a relatively small number of deployed honeypots can provide an effective defense strategy, often better than movement alone. Furthermore, the models confirm the intuition that that combining, or layering, defense mechanisms provide the largest impact to attacker success while providing a quantitative analysis of the improvement and parameters of each strategy.

1. INTRODUCTION

With the continuing advancement of cyber terrorism and digital warfare, a large emphasis is being placed on raising the security of all devices connected to the Internet. Researchers and security practitioners must seek to provide innovative and game-changing solutions to tip the scale back in favor of the defender in the ongoing arms race. One proposed tenant of an improved defense is to design a set of imperfect but challenging mechanisms that are layered and dynamic to increase the burden of mounting a successful attack in all aspects for an attacker. In this paradigm, the attacker can no longer make static, long-term assumptions about the state of the network, the devices or the human participants. This added complexity can frustrate the attacker, exhaust their resources and cause a suboptimal response or attacker that increases the likelihood of being detected and an unsuccessful attempt. For example, reconnaissance defenses are designed to hinder an attacker's progress at a very early stage, where the attacker's focus is to collect information about potential targets. This stage is critical for the attacker since it directly affects later stages and ultimately the success of the attack.

Reconnaissance defenses typically involve movement, deception, or a combination of both. Moving Target Defenses (MTDs) operate by constantly changing the attack surface. Network address shuffling is an example MTD that remaps addresses in an attempt to render scanning useless. In comparison, deception defenses attempt to mislead the attacker by mimicking different infrastructure components. For example, honeypots can be deployed to mislead attackers performing reconnaissance about potential targets in a network. Both reconnaissance defense types can invalidate any information gathered by the attacker as well as reveal critical information about the attacker and/or the attack strategy. Although deploying reconnaissance defenses is an effective defense strategy, especially when combined [1], there is typically a nontrivial administration cost in terms of both initial configuration and maintenance. For example, honeypots often require continual attention and maintenance that grows as the number of honeypots increases. Therefore given the benefit and cost associated with reconnaissance defenses, understanding under what circumstances they are most effective is important.

In this regard, this paper develops flexible theoretical models that describe the defense gains of deploying reconnaissance defenses. The probabilistic models are simple and can be directly and easily modified based on the assumptions regarding both the defense and attacker strategies. Since the model is probabilistic, the output is easily interpretable and can be directly applied to decisions regarding the deployment and placement of many reconnaissance defenses. We propose and evaluate two simple attacker scenarios, foothold establishment and minimum to win. Foothold attacks seek

MTD'15, October 12, 2015, Denver, Colorado, USA.

DOI: http://dx.doi.org/10.1145/808475.2808480.

to compromise a single vulnerable computer in hopes of leveraging it against other, possibly local, targets. In contrast, minimum to win attacks desire to exploit a minimum number of computers, for botnet development or to discover a secret distributed across multiple computers. The theoretical models described in this paper provide the necessary tools to demonstrate the effects that network size, quantity of vulnerable systems, quantity of addresses scanned, and the number of honeypots have on the probability attacker success. In addition, empirical studies are provided using actual network traces comparing the benefit of honeypots, address network shuffling, and combining honeypots and network address shuffling.

The remainder of this paper is structured as follows. Section 2 describes different reconnaissance cyber defenses, including movement and deception. Honeypot are described in more detail in Section 2.2, while Section 3 introduces probabilistic models for honeypot defense performance. Honeypot performance analysis is performed in Section 4, and simulation is used in Section 5 to compare movement and deception reconnaissance defenses. Finally, Section 6 summarizes reconnaissance defense performance models and findings, and provides some directions for future work.

2. RECONNAISSANCE DEFENSES

A cyber attack can be divided into a sequence of stages. As depicted in Figure 1, a simplified attack model typically starts with a reconnaissance stage that is followed by an associated attack attempt after a period of time has elapsed. The first stage, reconnaissance, involves the attacker seeking to gain intelligence about potential targets. This is often considered the most important stage since the subsequent attack stages rely heavily on the information obtained. As a result, it has been found that well-resourced adversary can spend approximately 45 percent of his time performing reconnaissance. Given the expense, in terms of time and resources, and reliance of this information, defending an attack at the reconnaissance stage can be a very successful strategy.

2.1 Movement Defenses

One category of cyber defense at the reconnaissance phase is a Moving Target Defense (MTD) [2, 3]. These defenses rely on *movement* to make the attack space appear different over time. For example, network address shuffling [4, 5, 6] prevents an attacker from reliably contacting a defended computer by periodically remapping network addresses. As a result, any address information that is obtained during reconnaissance is rendered useless if a shuffling event occurs before the attack begins. For example, hit-list worms and botnets attempting to spread rely on scans over large chunks of address space for locating vulnerable hosts where a shuffling of the active addresses can render these scans useless. While address shuffling can provide an effective defense [5, 6, 4], deployment and implementation costs can negate the benefit.

2.2 Deception Defenses

Another form of cyber defense at the reconnaissance phase is deception [3]. Deception is a mechanism that attempts to distort or mislead an attacker into taking a course of action that is more suited to the goals of the defender. A common deception defense is the use of network honeypots. A honeypot is a computer system that is designed to be a trap for unauthorized accesses [7, 8]. Honeypots are deployed within a network to appear like normal, active systems to an outsider. The deception technique being employed is mimicking [9]. A honeypot attempts to mimic a real system to fool the adversary into probing and/or attacking it. The amount of interaction the honeypot performs can vary [10]. High interaction honeypots respond to queries with information that represents a possible system within the infrastructure but unlike a normal system, it maintains very detailed logs of all interactions. From these detailed logs, administrators can gain insight into an attacker's goals and methods as well as put in place other measures to hopes of preventing an attack.

This research provides models that describe the theoretic bounds of the effectiveness of using honeypots as a deception tool.This is especially important given the nontrivial cost associated with honeypot deployment and their maintenance [8]. The models provide insight into the expected, long term outcomes of using honeypots. These models can then be used to guide honeypot deployment given the constraints and requirements of a particular network infrastructure.

3. RECONNAISSANCE PERFORMANCE MODELS

A primary objective of this paper the development of models the performance of reconnaissance defenses, specifically the probability that an attacker will be successful. A common tool used for modeling the probability of outcomes for a system is an urn. An urn is a simple vessel containing a set of marbles consisting of different colors, representing different outcomes or events. A player then draws a marble from the vessel, corresponding to a random selection according to some unknown distribution, and notates it's color and then proceeds to again. The urn model for determining the long term probabilities of a given system has been leveraged in physics, communications, and computer science [4]. Urns are useful as a modeling tool as they provide a concrete medium for determining statistical distributions given a set of events. They are used to estimate the probabilities of various outcomes based on various processes over time such as drawing a certain color marble or the number of expected draws before a certain color appears.

Consider a network infrastructure that consists of variety of vulnerable computers, secure computers, and honeypots. Specifically assume the following characteristics about the network.

- There are n total addresses available to the administrator (address space) and $v \leq n$ vulnerable computers.
- There are h honeypots within the network.
- The attacker has probability d of being deceived by a honeypot.
- The attacker is aware of the address space (n addresses) and will serially attempt k connections, $k < n$.
- Given k attempts, the attacker wins if they are able to locate m vulnerable computers within the network **without** contacting and being deceived by a honeypot.

Given these constraints, consider an urn consisting of n marbles, the number equals the number of addresses within the

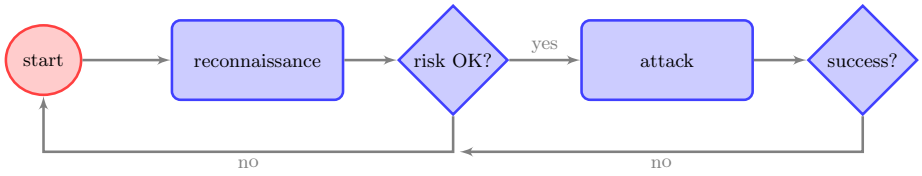

Figure 1: An abbreviated process diagram for a typical cyber attack.

network. Of the n marbles, assume v green marbles represent the number of vulnerable computers within the network, h red marbles represent honeypots, and $n-(v+h)$ blue marbles represent the remaining addresses within the network (devices/addresses of no value to the attacker). Note n is equal to the number of addresses, not the number of computers within the network.

The attacker can attempt k reconnaissance actions serially (one at a time) or in parallel (all k at once). These reconnaissance actions are modeled as either k individual draws (serial) or a single draw of k marbles from the urn (parallel). For example, if the attacker wanted to attempt a full scan of the network, then $k = n$. The attacker success depends on the outcome of the k draws, where the attacker must draw at least one green marble but no undetectable red marbles. The population of undetectable marbles (representing undetectable honeypots) is hd, where d is the probability the attacker is deceived by the honeypot. Therefore $d = 1$ represents the attacker is always deceived by the honeypot. The outcome of at least one green and no undetectable red marbles, represents discovering the location of key infrastructure and avoiding detection. The model will show the impact of honeypots by simply modifying the different marble populations in the urn. This will provide bounds regarding how well this type of defense theoretically performs in terms of the probability of the attack success.

3.1 Undefended Model

Consider the case where the network is not defended. As a result, the attacker can simply iterate through the network address space and gain perfect knowledge about the computers. If the attacker employs a $k = n$ strategy, their probability of success (drawing at least one green marble using the urn model) is one. Otherwise, if $k < n$, an urn model can be used to provide the probability of attacker success.

In the undefended case, consider a simplified urn model with n marbles, consisting of two populations. v green marbles and $n-v$ blue marbles. At each turn, the attacker draws a marble and does not replace it. The attacker continues to draw and not replace the marbles, k times. This process is termed a hypergeometric distribution, "number of successes in a sequence of k draws from a finite population without replacement" [11]. Note, the "without replacement" condition allows this model to apply to serial and parallel reconnaissance scans. Let X_k be a random number that follows the hypergeometric distribution for drawing x green marbles on k draws from the urn.

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

The hypergeometric distribution accurately models the attacker and the target network. With static addresses, any probe performed by the attacker will provide knowledge that will remain true, i.e. no reason to probe the same address again. Therefore, this maintained knowledge is equivalent to an urn model with non-replacement draws.

Given Equation 1, calculating the probabilities of the attacker success requires simple probabilistic manipulation. As previously stated, the attacker's success requires the discovery of one vulnerable computer, equivalent to drawing at least one green marble, $Pr(0 < X_k \leq v)$. The probability of drawing at least one green marble is,

$$Pr(X_k \geq 1) = 1 - Pr(X_k = 0) = 1 - \frac{\binom{n-v}{k}}{\binom{n}{k}} \tag{2}$$

The expected number of vulnerable computers discovered, i.e. green marbles drawn, when $k \leq n$ is,

$$E(X_k) = k\frac{v}{n} \tag{3}$$

3.2 Honeypot Defense Model

Using the urn model described in the previous section, it is possible to derive the chances of attacker success and failure (detection by administrators) when honeypots are used as a reconnaissance defense. With honeypots in place, if the attacker draws $k = n$ marbles and $h > 0$ (at least one honeypot exists), then the attacker has zero probability of success. The attacker would discover all vulnerable computers but would also draw at least one red marble representing a honeypot.

The probability of attacker success where $k < n$ can be derived using the multivariate hypergeometric distribution. A multivariate hypergeometric distribution describes the probability of drawing x marbles with k draws from a finite population consisting of more than two colors without replacement. Again, this model is applicable for serial and parallel reconnaissance scans due to the "without replacement" condition. In terms of modeling honeypots, the following is the multivariate hypergeometric distribution:

$$Pr(X_k = x) = \frac{\binom{v}{x}\binom{hd}{l}\binom{n-(v+hd)}{k-(x+l)}}{\binom{n}{k}} \tag{4}$$

Note the marble population hd represents the honeypots that always deceive the attacker. If $d = 0$ then a honeypot never deceives the attacker, while $d = 1$ represents a honeypot that always deceives the attacker. The parameter l represents the number of undetectable red marbles (undetectable honeypots) drawn. If the attacker cannot survive contacting any undetectable honeypots, then Equation 4 can be reduced to:

$$Pr(X_k = x) = \frac{\binom{v}{x}\binom{n-(v+hd)}{k-x}}{\binom{n}{k}} \tag{5}$$

Since there is no replacement in a hypergeometric distribution, the probabilistic mean, or expected value, is simply the number of draws, k, times the number of marbles of a particular color divided by the total population of the urn. The expected number of green marbles drawn, representing the number of vulnerable computers, given k draws is the same as for the undefended case (Equation 3). However now there is a chance of contacting a honeypot, and the expected number of red marbles, representing the number of honeypots, given k draws is,

$$E(X_k) = k\frac{hd}{n} \tag{6}$$

3.2.1 Allowable Losses

Equation 5 assumed the attacker lost if at least one honeypot was contacted within k reconnaissance attempts. However, a *well-resourced* and determined attacker may be willing to sustain at least L losses in order to find key infrastructure resources. For example, an attacker with access to a large botnet may be willing to lose L bots in return for the location of a key computer.

Equation 4 can be used to determine the attacker's success if losses are acceptable (contact a undetectable honeypot). The probability of attacker success is determined by summing the possible outcomes assuming no more that L losses are allowed within k reconnaissance attempts (serial or parallel):

$$Pr(X_k \geq 1 \text{ and } l \leq L) = \sum_{l=0}^{L}\sum_{x=1}^{k-1} \frac{\binom{v}{x}\binom{hd}{l}\binom{n-(v+hd)}{k-(x+l)}}{\binom{n}{k}} \tag{7}$$

3.3 Shuffling Defense Model

A probabilistic model for movement defenses, specifically network address shuffling, was originally introduced in [4]. A similar urn model was used to model *perfect shuffling*, where the administrator shuffles after every reconnaissance attempt. As a result, the attacker's knowledge between scan attempts is limited. Although perfect shuffling is difficult to implement in practice, the model provides an upper bound on the defense performance [4].

Consider an urn model where drawn marbles are returned. The attacker success probability can be 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

$$\Pr(X_k = x) = \binom{v}{x} p^x (1-p)^{k-x} \tag{8}$$

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

$$\Pr(X_k > 0) = 1 - Pr(X_k = 0) = 1 - (1-p)^k . \tag{9}$$

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 [11]. 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

$$E[Y] = \frac{1}{p} = \frac{n}{v}. \tag{10}$$

Using these equations along with a few assumptions, shuffling was found to be an acceptable defense if there was a small population of vulnerable systems within a large network address space [4]. However, shuffling has a cost for legitimate users since these connections may become lost once a shuffling event occurs. This aspect of shuffling is explored empirically in Section 5 of this paper.

4. ANALYSIS OF HONEYPOT DEFENSES

As done for network address shuffling [4], the analysis of honeypot defenses will be performed given two different attack goals. The first attack goal considered is the establishment of a *foothold* in the targeted network. In this situation the attacker wants access to a certain computer in the infrastructure; however, connecting directly to this computer is impossible. For example, the vulnerable machine may be protected by a firewall. A strategy for the attacker in this situation is to establish a foothold by compromising another computer or set of computers in the infrastructure that are accessible. The foothold inside the protected infrastructure can then be leveraged to gain access to the computer behind the firewall. This strategy is a lower risk for the attacker as their actions can be more direct and precise, attracting less attention with this less pronounced behavior.

Another possible goal for an attacker is to compromise a minimum number of computers (*minimum to win*) within the targeted network. This scenario exists if the attacker's goal is to gain information that is distributed across multiple computers or if the goal is to acquire as many resources as possible, for example for botnet recruitment [12]. As in the foothold scenario, the size of the network, the number of vulnerable computers and the number of scans will affect the probability of success for the attacker. In addition, the number of unique computers the attacker needs to compromise will also impact their probability of success.

4.1 Foothold Scenario

Given the attacker's desire to locate a single vulnerable computer, it is important to determine the effect of deploying honeypots within the network. The attacker will only succeed if a single vulnerable computer within the address space is located without contacting a honeypot. If the attacker contacts a honeypot, the attacker is discovered and is blocked from communicating with computers within the network. This constraint will be relaxed in Section 4.1.4, where the attacker is willing to contact a certain number of honeypots.

4.1.1 Number of Scans

An important variable in measuring the performance of honeypots as a defense is the effect of the number of scans attempted by the attacker. While an increased number of scans will lead to a higher chance of locating and securing a foothold, it also increases the likelihood of detection by a honeypot. For example, consider a class-C network (255 addresses) where 25% of the computers are vulnerable. Figure 2 shows the effect of the attacker increasing the number of scan attempts in this situation. If no honeypots exist

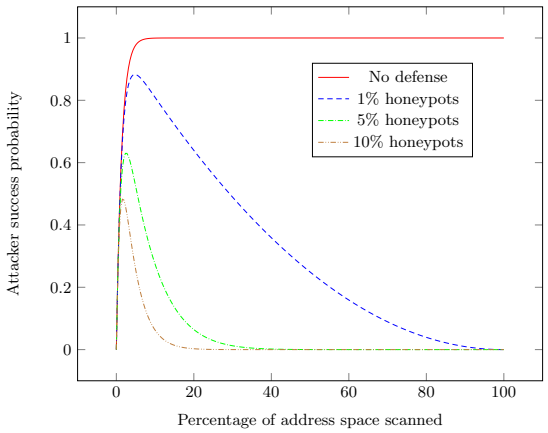

Figure 2: Average attacker success probability for gaining a foothold in a class-C network as the number of attacker scans increases. Different lines represent different percentages of honeypots deployed.

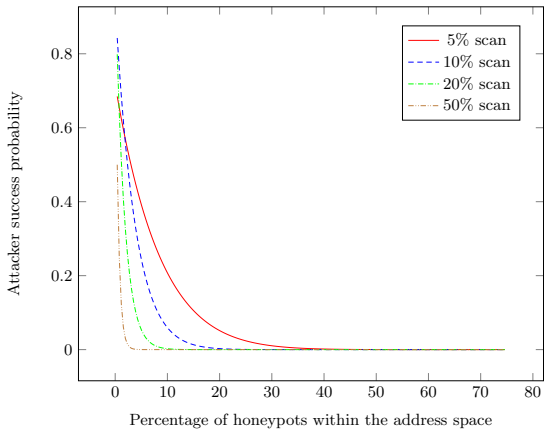

Figure 3: Average attacker success probability for gaining a foothold in a class-C network as the number of honeypots deployed increases. Different lines represent different attacker scan rates.

within the address space, the attacker can quickly find a vulnerable computer as discussed in Section 3.1. With 1% of the address space consisting of honeypots (3 honeypots), the probability of success increases until the scan rate reaches 5% of the address space. After this scan rate, the attacker is more likely find a honeypot within the address space. Their probability of probing a honeypot increases which will cause the probability of attacker success to decrease as the scan rate approaches 100%. If the number of honeypots deployed increases to 5% (or 12 honeypots) the success probability of the attacker drops substantially, where the attacker has a maximum success rate of 62% percent with a 2% scan rate. With 10% honeypots the maximum success rate for the attacker falls below 50%. This provides evidence that the presence of honeypots has a significant effect on the probability of detection.

4.1.2 *Number of Honeypots*

Given that the goal of the attacker is to locate a single vulnerable computer without contacting a honeypot, the probability of success for the attacker is impacted by the number of honeypots deployed within the network. Therefore, if there are no honeypots and the attacker scans the the entire network, the the attacker is assured to contact all the vulnerable computers. If at least one honeypot is deployed under the same circumstances, the attacker will never be successful since they will always contact a honeypot. If they contact a honeypot before reaching the k^{th} scan, the attacker will continue to scan. This assumption is made to simplify the probability model. It also models reality in the case where the attacker's reconnaissance is automated, without carefully observing the result after each scan event. Therefore, contacting a honeypot would not cause the attacker, or script, to stop scanning. To analyze the impact of different honeypots, the number of scans within the network must be $k < n - hd$. If not, the probability of attacker success is zero following the same logic.

Using the model established in Section 3.2, the attacker's probability of success in the given scenario can be determined using Equation 4. As an example, again consider a class-C network where 10% of the addresses are vulnerable computers. The probability of attacker success as the percentage of honeypots to number of addresses increases is shown in Figure 3. As expected, the probability of attacker success when there are no honeypots within the space is proportional to the scan rate, which is equivalent to no defense discussed in Section 3.1. The figure demonstrates that increasing the number of honeypots does cause the probability of success to quickly decrease. This trend is also independent of the number of scans performed, honeypots within the network address space increase the probability of being detected.

4.1.3 *Honeypot Deception*

Performance analysis thus far has assumed the attacker is unable to distinguish a real device from a honeypot; however, techniques do exist for determining if a device is a honeypot [13, 14]. The impact of the attacker's ability to detect a honeypot can be observed given a class-C network where 10% of the addresses are vulnerable computers and the attacker is able to scan 10% of the address space. Figure 4 depicts the probability of attacker success in this scenario as the attacker is increasingly likely to be deceived by a honeypot. When honeypots are deployed, the probability of attacker success consistently decreases as the likelihood of attacker deception increases. The reduction is increasingly sub-linear with larger populations of honeypots. If no defense is provided, the attacker success probability is 93%. With 5% honeypots deployed, the attacker is success rate drops to 50% if the detection rate is 50%. In contrast, with 10% honeypots the success rate drops to 25% if the detection rate is 50%. This is expected since the population of undetectable honeypots is increasing, thus providing a greater defense capability. This is also shown in Figure 3, where an increasing number of honeypots has a similar non-linear impact on the attacker success rate.

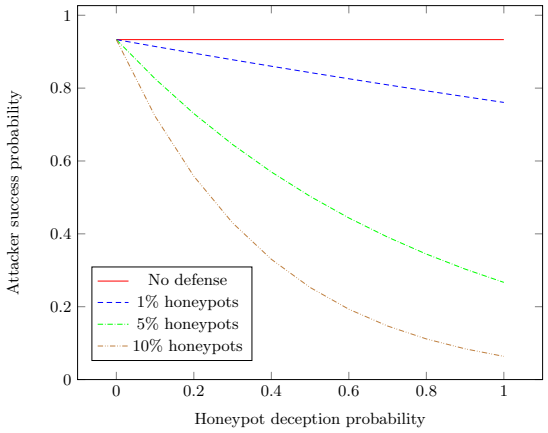

Figure 4: Average attacker success probability for gaining a foothold in a class-C network as the honeypot deception probability increases. Different lines represent different percentages of deployed honeypots.

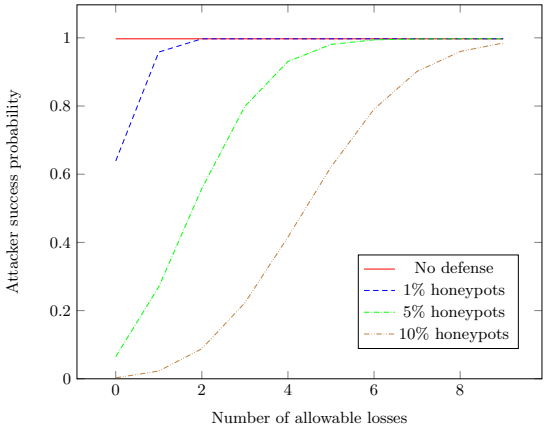

Figure 5: Average attacker success probability for gaining a foothold in a class-C network as the number of allowable losses increases. Different lines represent different percentages of deployed honeypots.

4.1.4 *Allowable Losses*

The previous foothold examples assumed the attacker lost if at least one honeypot was contacted within k reconnaissance attempts. However as mentioned in Section 3, an attacker may be willing to sustain losses (network address blocking) if the target is sufficiently desirable. For example, if the attacker owns a large botnet, then losing a certain number of bots may be inconsequential.

As an example of allowable attacker losses, consider a class-C network where 10% of the addresses are vulnerable computers and the attacker can scan 10% of the addresses space. The probability of attacker success as the number of allowable losses increases is shown in Figure 5. The success rate of the attacker increases as the number of allowable losses increases. The maximum success rate is achieved once the number of allowable losses is equal to pre greater than the number of deployed honeypots. The figure demonstrates that well-resourced and determined attacker has a high success rate given an allowable number of losses.

4.2 Minimum to Win Scenario

The previous scenario considered an attacker searching for single computer with the address space modeling a foothold attack scenario. In that example, an attacker has a reasonable chance of contacting at least one vulnerable computer when there is no defense present. Consider another scenario where there are multiple vulnerable computers within the address space. However now assume the attacker must find more than one vulnerable computer. The motivation for the attacker could be the need to compile a large number of resources for a botnet or the information they require is distributed across many nodes within the infrastructure.

Given this new scenario, the number of vulnerable computers will be $v \geq 1$. As in the previous section, Equation 4 will be used to calculate v the probability of attacker success. The substitution for the new scenario changes equation slightly, instead of only looking at the number of ways of drawing $k-1$ blue marbles and at least one green marble, the number of ways of pulling at least x green marbles from the total green population must be factored into the probability. As before, the number of undetectable red marbles drawn, representing honeypots that decieve the attacker, must remain below the allowable loss threshold.

4.2.1 *Number of Scans*

An important parameter for analyzing attacker scans. Figures 6 and 7 provide some insight about the relationship between the attacker success probability and scanning rate for minimum to win scenarios. Consider a class-C network where 25% of the addresses are associated with vulnerable computers. In addition, assume the attacker must contact at least 25% of the vulnerable computers (at least 16 of the 64 vulnerable computers). Figure 6 demonstrates the probability of attacker success as the network address scan percentage increases. The case of no defense is similar to Figure 2, except there is a zero probability of success until the attacker is able to make at least 16 scans. The addition of honeypots decreases the probability of attacker success because the attacker becomes more likely to locate a honeypot as well as the vulnerable computers necessary for success. Even deploying 1% honeypots has a dramatic effect.

Figure 7 depicts another interesting aspect of minimum to win scenarios. In this case 5% of the addresses are honeypots, 25% of the addresses are vulnerable, and the minimum percentage of vulnerable computers required by the attacker ranges from 5% to 25% of the vulnerable. As the minimum number of vulnerable computers required increases, the success rate decreases. This shows the difficulty of locating an increasing minimum number of vulnerable computers. Again, placing a small number of honeypots within a network can provide a large benefit for defense.

4.2.2 *Number of Honeypots*

Given that the attacker is motivated to locate a larger set of computers within the minimum to win scenario, the number of honeypots is another parameter to consider. As an administrator, determining the number of honeypots to deploy to achieve a certain level of security could be a ben-

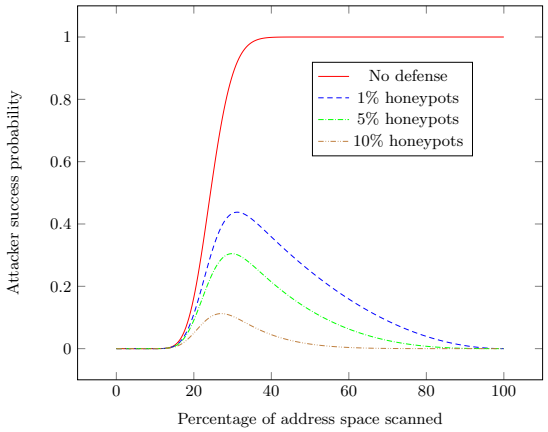

Figure 6: Average attacker success probability of obtaining at least 16 of the 64 vulnerable computers (minimum to win) in a class-C network as the number of attacker scans increases. Different lines represent different percentages of honeypot deployed.

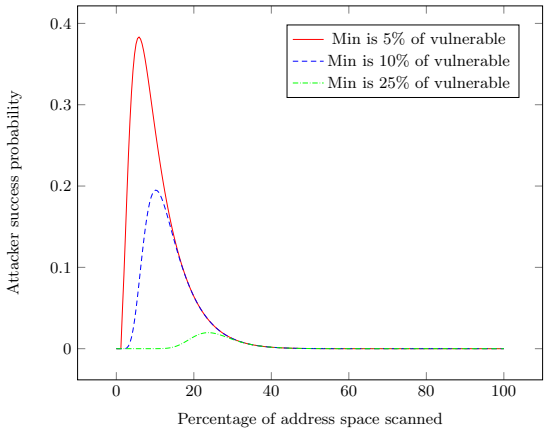

Figure 7: Average attacker success probability of obtaining a minimum percentage of the 64 vulnerable computers (minimum to win) as the number of attacker scans increases. The network had a class-C address space and the 5% of the addresses were honeypots. Different lines represent different minimum to win percentages.

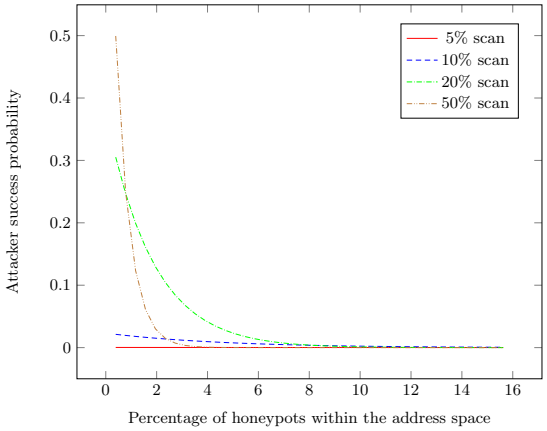

Figure 8: Average attacker success probability of obtaining at least 6 of the 25 vulnerable computers (minimum to win) in a class-C network as the number of honeypots deployed increases. Different lines represent the attacker scan rate.

eficial tool in preserving time and resources. Figure 8 shows that honeypots do have a significant impact on the success of the attacker and the size of scan they can attempt without being detected. In this example, a class-C network has 10% vulnerable computers and the attacker is required to contact at least 25% of these vulnerable computers (at least 6 of the 25 vulnerable). The probability of success decreases dramatically at a scan rate of 50% as the number of honeypots increases only slightly. Probabilistically, there is slightly less of an immediate impact on lower scan rates. However, as previously noted, a small percentage of deployed honeypots is an effective defense against the minimum to win attack scenario.

4.2.3 Number of Vulnerable Computers

The minimum to win strategy introduces a new parameter in determining the probability of attacker success with the model described in Section 3. Given that the attacker's motivation is to locate a minimum number of vulnerable computers, the probability of success is affected by the number of vulnerable computers, v. For example, consider a class-C network and the attacker seeks to contact at least 16 computers in 64 scans (25% of the addresses). The probability of success when there are no honeypots is the same as the no defense case.

5. COMPARING AND COMBINING RECONNAISSANCE DEFENSES

As introduced in the Section 2, network address shuffling and honeypots are examples of reconnaissance defenses. This section will compare the performance of shuffling, honeypots, and their use in combination. Theoretical performance models do exist for the use of network address shuffling in isolation under certain assumptions about the shuffling epoch [4]; however, modeling shuffling under more realistic attacker scenarios where the attacker needs to both locate and reliably compromise a set of machines is more challenging.

A discrete event simulator was created to collect empirical data. For normal traffic traces and connection information, we utilize the 2008 SIGCOMM conference packet traces, using the flow start and end times for both normal and attacker connections for a class-C network, 255 addresses [15]. These traces are primarily web-oriented, with the majority of the flows consisting of DNS or HTTP. The traffic consists of over 1,000 unique destination IP addresses, of which 255 were randomly chosen for each simulation, utilizing their corresponding flows to provide the start and end of normal connections. Addresses were then assigned one of three types, normal, vulnerable or honeypot, signifying which computers an attacker can compromise or be detected by. For the simulations 10% of the addresses contained vulnerable computers, while 4% of the addresses were designated as honeypots.

To simulate a more realistic attack scenario, an attack is divided into a reconnaissance phase followed by an attack phase. The attacker is allowed to attempt k scans during the reconnaissance phase. The attacker will then attempt to compromise any vulnerable computers discovered during the first phase after waiting 10 minutes, termed the *attack wait time*. An attack is deemed successful if a vulnerable computer is contacted during both phases. When honeypots are used as a defense, the attacker fails if a honeypot is contacted in either phase. When address shuffling is used as a defense, the attacker fails if reconnaissance reveals a vulnerable computer that cannot be subsequently contacted during the attack phase due to a shuffle event. Each experiment was simulated 1000 times to provide a stable sample of the attacker success rates under each defense mechanism.

5.1 Foothold Scenario

As described in section 4, the foothold attack scenario requires the attacker contact at least one vulnerable computer. Experiments were performed with different shuffling rates relative to a specified attacker wait time (amount of time between reconnaissance and attack phases) as done in [4]. Results reference the ratio of inter-shuffle time to attack wait time; therefore, a ratio of 2 indicates the inter-shuffle time is twice as long as the attacker wait time. For these experiments the attacker was allowed to scan 10% of the network.

As seen in Figure 9, the performance of the shuffling defenses were directly affected by the shuffle time. As the shuffle time increased (indicated on the graph by larger ratios), the attacker success rate also increased since the likelihood of a shuffle event occurring between reconnaissance and attack was less likely. Once the shuffle time was greater than the attack wait time, the honeypot-only defense performed better than the shuffle-only defense. The attacker success rate with the honeypot-only defense remains constant as predicted by Equation 4. The combination of shuffling and honeypots provided the best defense, since both are capable of defending against an attack. As the shuffle time increased the defensive effect of shuffling decreased and as expected, and the performance of the combined defense approached a honeypot-only defense.

5.2 Minimum to Win Scenario

The minimum to win scenario requires the attacker to contact a certain percentage of vulnerable machines in a network. Instead of considering different inter-shuffle times, these experiments focused on varying the minimum percentage of vulnerable computers required for attacker success. Therefore the inter-shuffle time was 20 minutes.

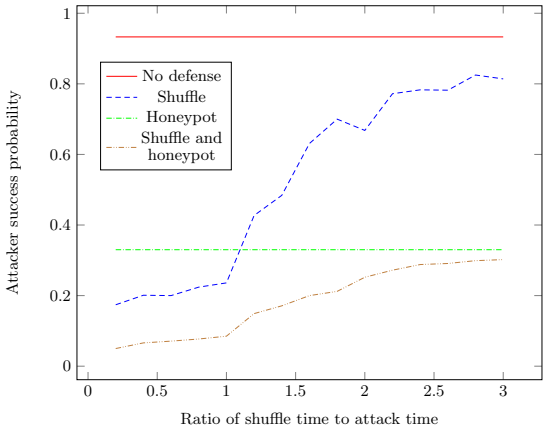

Figure 9: Average attacker success probability for gaining a foothold in a simulated class-C network as the ratio of shuffle time to attack increases. Different lines represent different defenses.

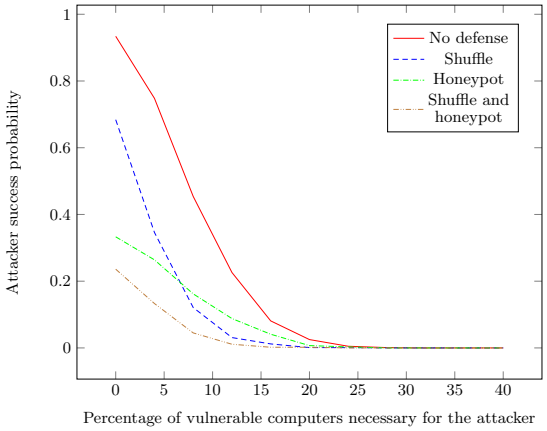

Figure 10: Average attacker success probability as the minimum number of vulnerable computers required increases in a simulated class-C network. Different lines represent different defenses.

The performance of the different defense strategies for the minimum to win scenario is given in Figure 10. For all types of defenses, as the percentage of vulnerable computers required for the attacker to win increased the success rate decreased. At low minimum to win percentages, shuffling yielded higher attacker success rates as compared to the honeypot defenses. This is expected since the shuffle time is twice the attack wait time. However, once more than one vulnerable computer was required for the attacker to win (8% of the vulnerable), shuffling performed slightly better than the honeypot-only defense. This is primarily due to shuffling potentially affecting either attack phases.

5.3 Shuffling Drop Probability

While network address shuffling does provide a defense against reconnaissance, there is the potential to disconnect

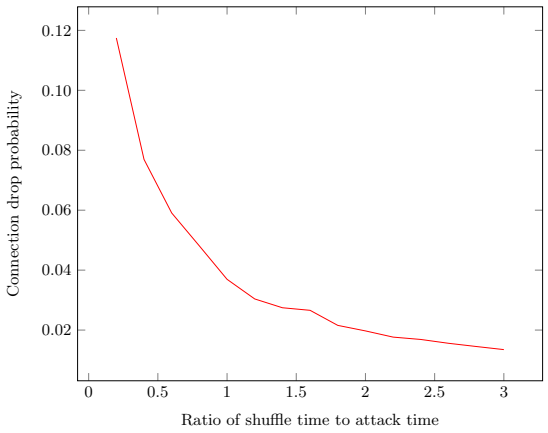

Figure 11: Connection drop probability as the shuffle time increases.

legitimate connections [4]. Shuffling less frequently reduces the drop probability for legitimate connections; however as shown in the previous experiments, shuffling is most effective when the inter-shuffle time is less than the attack wait-time.

As seen in Figure 11, for these traffic traces the drop probability for legitimate connections is approximately 0.04 when the inter-shuffle time equals the wait time. As the inter-shuffle time increases, the drop probability continues to decrease as expected. However, these longer inter-shuffle times are also associated with higher attacker success rates, as seen in Figure 9. Therefore as discussed in [4], finding the correct inter-shuffle time required some knowledge of the attack wait and average legitimate connection time. According to Rowe and Goh, attackers can wait up to a day before acting on reconnaissance information [16]. Then it would be reasonable to have network shuffles only a few times a day, limiting the overhead of shuffling while interrupting a very small percentage of daily network traffic. Determining the best time for a shuffle is important as well (i.e., shuffling during business hours may be too disruptive). The addition of a few honeypots can further reduce the shuffling frequency (thus increasing inter-shuffle period) necessary for a desired attacker success rate.

6. CONCLUSIONS

This paper introduced a set of probabilistic models that can help administrators deploy honeypots given basic infrastructure characteristics. For example, it was observed that a relatively small number of honeypots can offer a significant cyber defense in many situations. The defense performance of honeypots (deception) was typically better than network shuffling (movement). However, it was found that the combination of deception and movement provided the best reconnaissance defense performance.

The models introduced can help provide direction for honeypot deployment, and can serve as a framework to address additional questions. For example, reconnaissance defenses can have a high administrative cost. The defense performance models described in this paper can be combined with cost models to provide better deployment guidance. The inclusion of cost can help determine what level of honeypot deployment is worth the operational expense.

7. REFERENCES

[1] F. Cohen and Associates, "Moving target defenses with and without cover deception," Downloaded from http://all.net/Analyst/2010-10.pdf, Oct. 2010.

[2] J. H. H. Jafarian, E. Al-Shaer, and Q. Duan, "Spatio-temporal address mutation for proactive cyber agility against sophisticated attackers," in *Proceedings of the First ACM Workshop on Moving Target Defense*. ACM, 2014, pp. 69–78.

[3] N. C. Rowe, "Measuring the effectiveness of honeypot counter-counterdeception," in *Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS)*., vol. 6. IEEE, 2006.

[4] T. E. Carroll, M. B. Crouse, E. W. Fulp, and K. S. Berenhaut, "Analysis of network address shuffling as a moving target defense," in *Proceedings of the IEEE International Conference on Comminucations*, 2014.

[5] 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.

[6] L. Shi, C. Jia, S. Lü, and Z. Liu, "Port and address hopping for active cyber-defense," in *Intelligence and Security Informatics*, ser. Lecture Notes in Computer Science. Springer, 2007, vol. 4430, pp. 295–300.

[7] N. C. Rowe, "Measuring the effectiveness of honeypot counter-counterdeception," in *Proc. of the 39th Annual Hawaii Int. Conf. on System Sciences (HICSS '06)*, 2006, p. 129.3.

[8] L. Spitzner, "The honeynet project: trapping the hackers," *Security & Privacy, IEEE*, vol. 1, no. 2, pp. 15–23, Mar 2003.

[9] J. F. Dunnigan and A. A. Nofi, *Victory and Deceit: Deception and Trickery at War*, 2nd ed. San Jose, California, USA: Writers Club Press, 2001.

[10] R. McGrew, "Experiences with honeypot systems: Development, deployment, and analysis," in *Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS)*, vol. 9, 2006.

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

[12] 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)*. Springer, 2004, pp. 258–277.

[13] X. Fu, W. Yu, D. Cheng, X. Tan, K. Streff, and S. Graham, "On recognizing virtual honeypots and countermeasures," in *Proc. of the 2nd IEEE Int. Symp. on Dependable, Autonomic and Secure Computing (DASC '06)*, 2006, pp. 211–218.

[14] T. Holz and F. Raynal, "Detecting honeypots and other suspicious environments," in *Proceedings of the 2005 IEEE Workshop on Information Assurance and Security*, 2005.

[15] "Sigcomm 2008 network traces," http://www.cs.umd.edu/projects/wifidelity/sigcomm08_traces/.

[16] N. C. Rowe and H. C. Goh, "Thwarting cyber-attack reconnaissance with inconsistency and deception," in *Information Assurance and Security Workshop, 2007. IAW'07. IEEE SMC*. IEEE, 2007, pp. 151–158.