Towards Cost-Effective Moving Target Defense Against DDoS and Covert Channel Attacks

Huangxin Wang
Department of Computer Science
George Mason University
Fairfax, VA 22030
hwang14@gmu.edu

Fei Li
Department of Computer Science
George Mason University
Fairfax, VA 22030
lifei@cs.gmu.edu

Songqing Chen
Department of Computer Science
George Mason University
Fairfax, VA 22030
sqchen@cs.gmu.edu

ABSTRACT

Traditionally, network and system configurations are static. Attackers have plenty of time to exploit the system's vulnerabilities and thus they are able to choose when to launch attacks wisely to maximize the damage. An unpredictable system configuration can significantly lift the bar for attackers to conduct successful attacks. Recent years, moving target defense (MTD) has been advocated for this purpose. An MTD mechanism aims to introduce dynamics to the system through changing its configuration continuously over time, which we call *adaptations*. Though promising, the dynamic system reconfiguration introduces overhead to the applications currently running in the system. It is critical to determine the right time to conduct adaptations and to balance the overhead afforded and the security levels guaranteed. This problem is known as the *MTD timing problem*. Little prior work has been done to investigate the right time in making adaptations. In this paper, we take the first step to both theoretically and experimentally study the timing problem in moving target defenses. For a broad family of attacks including DDoS attacks and cloud covert channel attacks, we model this problem as a renewal reward process and propose an optimal algorithm in deciding the right time to make adaptations with the objective of minimizing the long-term cost rate. In our experiments, both DDoS attacks and cloud covert channel attacks are studied. Simulations based on real network traffic traces are conducted and we demonstrate that our proposed algorithm outperforms known adaptation schemes.

1. INTRODUCTION

In traditional defense mechanisms, cyber systems are considered static. Attackers often have plenty of time to explore the vulnerability of these systems and maintain the gained privileges for extended periods of time. The static model disadvantages the defenders since they have to protect all the assets at all the time, while the attackers have the flexibility to choose when and what to exploit [4]. Consider an instance of cloud covert channel attacks [29]: Modern cloud providers employ static mechanisms to allocate users' virtual machines to physical hosts, and as long as a virtual machine is placed on a physical host, it is seldom reallocated or migrated to other physical hosts. As a result, once the attackers are co-resident (via techniques shown in [29]) with the defenders' virtual machines, they can decide when to conduct covert channel attacks as they like. The defenders need to protect their virtual machines against such attacks all the time through intensive system activity and traffic analysis [18].

A moving target defense (MTD) mechanism can be designed to mitigate the asymmetric advantage of attackers and raise the bar for their successful attacks. This mechanism essentially dynamically changes multiple dimensions of a protected system, and thus increases the complexity and uncertainty of discharging the attack target, reduces the attackers' windows of opportunity, and increases their cost of probing and attacking [16].

Most of the existing research in MTD has focused on developing different MTD techniques to change the static nature of the protected system. These techniques work by continuously/periodically changing one or several of the system/network configurations, e.g., network address, memory layout, execution environment, etc. An action of changing the system configuration state is called an *adaptation* [31]. Not exclusively, the proposed adaptation techniques include (1) address space layout randomization [4] — changing the memory address space layout dynamically to avoid pre-coded buffer-overflow attacks; (2) IP-Hopping [10] — changing a host's IP address dynamically to increase the complexity of network attacks such as communication eavesdropping and hijacking; and (3) execution environment adaptations [13, 27] — changing the software (such as OS) and/or hardware dynamically to make the execution environment different from time to time. However, adaptations come with overhead for the applications running in the protected system. Frequent adaptations can lead to significant performance degradation, while insufficient adaptations cannot effectively hinder potential attackers. The problem of determining the right time to make an adaptation is still in its infancy and it is formally defined as the MTD *timing problem* [31]. Most of the existing work that studies the effectiveness of various MTD mechanisms simply takes an approach of making these adaptations happen periodically [32, 13].

MTD'16, October 24 2016, Vienna, Austria

DOI: http://dx.doi.org/10.1145/2995272.2995281

To understand the importance of making timely adaptations in MTD mechanisms, we illustrate three defense scenarios in Figure 1. Figure 1 gives an overview of how an MTD mechanism works in thwarting attacks. Usually, attackers need to go through at least three phases in conducting a successful attack [22], including a probing phase, a constructing phase, and a launching phase. The three defense scenarios shown in Figure 1 are moving target defense mechanisms (1) without adaptations, (2) with *smart adaptations*, and (3) with *blind adaptations*, respectively. From this figure we can see that both smart adaptations and blind adaptations can effectively thwart the attacks by making adaptations before an attack is launched. However, blind high-frequency adaptation is myopic as it does not take into account of the power or frequency of attackers. Besides, it can be a burden for the host's applications especially when it is expensive to make adaptations. Thus, determining the right time to make adaptations is critical in making MTD more cost-effective and in reducing the service overhead brought by adaptations.

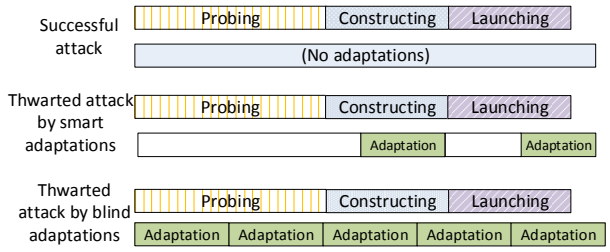


Figure 1: Various timing decisions in making adaptations in MTD

Recently, the MTD timing problem has started to catch researchers' attention. In [6], DeLoach *et al.* handled the MTD timing problem through relying on vulnerability scanning tools and intrusion detection tools to trigger alerts in making adaptations. Unfortunately, as pointed in [20], intrusion detection techniques are not guaranteed to detect attacks in a timely and accurate manner, especially when the attackers become smarter and more powerful. When a trustful intrusion detection solution is unavailable or when the probability of identifying intrusion detections is low, the defender is very likely to be attacked. Besides, this mechanism [6] does not take into account the adaptation cost and the potential damage — it cannot trade off security risks and costs.

In this paper, we take the first step to both theoretically and experimentally study the MTD timing problem. We aim to answer the question of *when is the best time to make economical adaptations*. We consider both *attacked cost* (the cost paid by a defender if it is under an attack) and *adaptation cost* (the cost paid by a defender if it makes an adaptation). Our proposed algorithm does not rely on any intrusion detection techniques to trigger alerts in making adaptations. Our objective is to design an adaptation strategy that guides the moving target defense mechanism to make adaptations in the right way to minimize the overall cost rate. We solve the MTD timing problem using the renewal reward theory to trade off both the adaptation costs and the attacked costs.

The rest of the paper is organized as follows. Section 2 introduces the problem model and describes our objective. Section 3 presents our designed algorithm. We conduct experiments to evaluate the proposed algorithm and compare it with up-to-date solutions in Section 4. A case study is further conducted to evaluate the proposed algorithm against cloud covert channel attacks in Section 5. Section 6 discusses the related works and concluding remarks are given in Section 7.

2. THREAT MODEL

In this section, we describe a threat model to characterize the moving target defense system and mechanisms. This theoretical framework follows the state-of-the-art MTD model proposed by Zhuang *et al.* [31]. Specifically, we describe an attack-defense scenario in which defenders employ MTD mechanisms, while attackers exploit the vulnerability of defenders to launch attacks.

Defenders.

A defender is regarded as an attack target for attackers. A defender could be one or more computer devices, programs, etc., which try to protect themselves from *confidentiality, integrity and availability (CIA) attacks*. A defender can be described by a set of configurations including its network addresses, its operating systems, etc. The set of configurations of a defender is denoted as *information parameter* [31]. The objective of the defender is to prevent the attackers from learning these information parameters. For example, a defender may try to hide a server's network address from the attackers to avoid distributed denial of service (DDoS) attacks.

Using moving target defenses, a defender can make adaptations to change the information parameters. For each adaptation, it incurs an *adaptation cost*. An adaptation cost includes the (amortized) cost of software/hardware required for making adaptations, such as purchasing operating system software, renting a pool of IP addresses, and the cost of perhaps temporarily slowing down the system performance during the adaptations.

Attackers.

An attacker could be one or several persons or automated code with the objective of attacking the CIA of the attack target. An attacker always needs to learn some critical/sensitive information parameters about the defenders in order to launch a successful attack. To gain knowledge of the defenders, an attacker usually conducts a *probing procedure* which may need time and computing or monetary resources from the attacker. Once an attack is successfully launched, an *attacked cost* incurs to the defender. The attacked cost is related to the information parameter of the defender as well as the attack types. For example, an attacked cost that a defender would suffer from a DDoS attack will be related to the amount of clients it is serving when the attack happens and the revenue rate [7].

In studying the MTD timing problem, we study a family of attacks. In such *attack-defend* setting, the probing time for an attacker to launch a successful attack, after a defender makes an adaptation, satisfies identical and independent distributions (i.i.d). This broad family of attacks includes covert channel attacks in cloud [18], IP address scanning and stealthy port scanning [10], and password cracking [26], etc.

Objective.

The objective of studying the MTD timing problem is to investigate the most economical way for a defender to make adaptations, taking into account of both the adaptation cost and the attacked cost. It is quite reasonable for a defender to endure risks of being attacked, if the adaptation cost is high while the attacked cost is low. We seek to answer the question of when is the best time to make an adaptation in order to minimize the long-run cost rate.

3. ALGORITHM

In this section, we first describe a moving target defense framework for which an algorithmic approach to the MTD timing problem can work on top of that. Then we present our algorithm in detail, which effectively decides the cost-effective adaptation time for defenders.

3.1 An MTD framework

A moving target defense framework which dynamically makes cost-effective adaptations is shown in Figure 2. The framework captures the defender's system, the adaptation cost, the attacked cost analysis components, the attack interval distribution fitting component, as well as the adaptation analysis engine. We walk through each component and their interactions in greater details in the following.

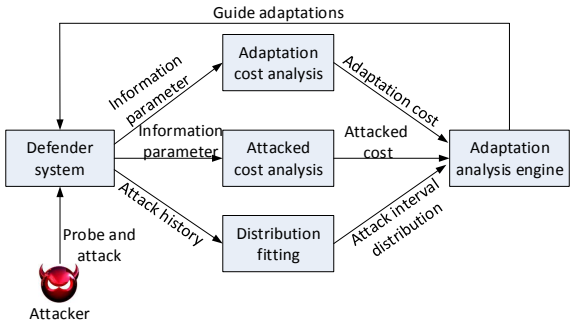


Figure 2: Overview of the MTD framework

Defender system.

A defender system has the objective of protecting itself from being attacked in a cost-efficient manner. In order to achieve the objective, it makes adaptations under the guidance of the *adaptation analysis engine*.

Adaptation cost analysis.

Adaptation cost analysis component measures and calculates the adaptation costs based on the information parameters of the defenders. An adaptation cost is the overall cost incurred to the defender system when it goes from one configuration to another.

Attacked cost analysis.

Attacked cost analysis component measures and calculates the attacked costs based on the information parameters of the defenders, as well as the applications currently running in the defender system. In general, the attacked costs should be calculated in an online manner since the set of applications may vary over time and so does its cost.

Distribution fitting.

The distribution fitting module fits the historical attack intervals to find out a distribution pattern of the attacks. It provides information about the distribution of the attack time intervals. This information is useful and critical in calculating the appropriate adaptation time to balance the adaptation cost and attacked cost for a defender. Note that this distribution may not necessarily satisfy a predefined probability distribution, but it should be identical and independent distributed.

Adaptation analysis engine.

The adaptation analysis engine is to predict the time to make the next adaptation. It takes the adaptation cost, the attacked cost, and the historical attack time intervals into account, and decides the most cost-effective time to make adaptations. The decision is made by formulating the adaptation problem as a renewal reward process and we solve the problem using the renewal reward theory [19]. The details of the algorithm are discussed in Section 3.3.

3.2 An example of attacks fitting in the framework

In order to illustrate how the proposed moving target framework works, we provide a concrete example of attacks. This example also shows the necessity of employing moving target defense.

Nowadays many cloud providers, such as Amazon EC2, Google Cloud and so on, allow multi-tenancy in which virtual machines from different customers are placed on the same physical machine. As a result, covert channel attacks have been identified as a threat to the current cloud environments [18, 29, 23]. Note that one of the main vulnerabilities of the clouds that lead to covert channel attacks is the static nature of virtual machine placement policy. Through covert channel attacks [18, 29, 23], a malicious virtual machine can steal/leak information from the virtual machines which are co-resident on the same physical machine. Leaking confidential information could harm the reputation and the profit of a cloud provider, making it lose customers.

Potential covert channel attackers work as below. As the cloud providers use static virtual machine placement policies in allocating virtual machines to physical hosts, once a customer's virtual machine is placed on a physical machine, it will seldom be reallocated to other physical machines. As a consequence, the attackers have plenty of time in finding the target virtual machine and achieving co-resident. Specifically, the attackers can exhaustively search for the target defender's virtual machines by launching virtual machines again and again until one of the virtual machines is allocated to the same physical machine as the victim virtual machine [29].

The moving target defense works as a counter-measurement to mitigate the covert channel attacks. The idea is to dynamically change the physical host for the target virtual machines to be protected, thus it increases the difficulty for the attackers to find the target victim virtual machines. Even if an attacker is lucky enough to be co-resident with the target victim at some time, the forthcoming moving (adaptations) of the victim's virtual machine will soon make the attacker lose his gained privileges (i.e., co-residence) over the victim, and thus stop the potential covert channel attacks.

Using the MTD model, we model the moving target defense against covert channel attacks in the cloud in the following. When a defender launches a virtual machine in the cloud, the virtual machine will be placed in a physical machine based on the static placement policy defined by the cloud provider. An adaptation migrates the virtual machine from one physical machine to another. The adaptation cost is measured using the migration cost based on the virtual machines' image sizes which affect the migration time and thus the system (including all applications running on the virtual machine being migrated) downtime. The attacked cost is the cost that a defender suffers from being attacked. For covert channel attacks, the attacked cost is measured using the payoff of information leakage. The objective of covert channel attackers is to achieve co-residence with the defender's virtual machines.

As attackers have no control over the cloud provider, the only action they can do to achieve the objective is to launch virtual machines using the cloud provider's services. As each time an attacker launches a virtual machine, the virtual machine may have a possibility of being assigned to the same physical machine where the victim virtual machine lies on, the attacker then can achieve the goal by launching virtual machines repeatedly until it finds co-residence with the victim virtual machine [29]. Adaptations made by the defender will change the physical machines that a virtual machine locates in, thus the co-residence privilege gained by the attacker will be erased. The attacker has to restart the process of probing to achieve co-residence in order to launch covert channel attacks.

3.3 A cost-effective adaptation algorithm

Let ψ denote the set of sensitive information parameters that a defender aims to protect. Let C_{adp} denote the adaptation cost. For example, in covert channel attacks, the adaptation cost is defined as the cost incurred by migrating the target virtual machine's image from the source host to the destination host. The value C_{adp} can be pre-calculated, if the set of various image sizes and the cloud's topology/links are known beforehand. Let C_{att} denote the attacked cost that a defender would suffer. The value C_{att} can be calculated in a periodical time manner. Note that calculating the values C_{att} does not imply that an adaptation should take place, even though if these values are high. Let Adp_{hist} denote the list of time intervals between two consecutive adaptations that we have so far. Let *t_elapsed* denote the current elapsed time since the last adaptation. The algorithm is designed to determine T, the time to wait for the next adaptation in order to minimize the overall cost in a long-term process.

We name our algorithm RRT (standing for renewal reward theory-based solution). RRT aims to determine the optimal adaptation time in the long-term process, subject to the above described input information. The procedure of this cost-effective adaptation algorithm is shown in Algorithm 1. Algorithm 1 illustrates that the moving target defense is a long-term process (as shown in the while loop in Line 1), and after each adaptation of the defender, the algorithm decides when to make the next adaptation in order to be cost-effective by calling a procedure called DAT (see Algorithm 2).

We remark Algorithm 1 here. Line 2 is the step of measuring the adaptation cost and attacked cost of the defender.

Algorithm 1 Cost-effective adaptation algorithm RRT

```
1: while TRUE do
2:     measure C_adp, C_att from ψ and the defender system;
3:     T = Decide-adaptation-time(C_adp, C_att, Adp_hist);
4:     t_elapsed = 0;
5:     while t_elapsed < T do
6:         if being attacked then
7:             break;
8:         else
9:             update t_elapsed;
10:        end if
11:    end while
12:    make adaptations;
13:    update Adp_hist = Adp_hist ∪ min{T, t_elapsed}.
14: end while
```

Line 3 makes decisions for the next adaptation time using the *decide-adaptation-time* (DAT) procedure described in Algorithm 2; Lines 5-13 indicate that if the attack happens before the derived cost-effective adaptation in Line 3, then the defender needs to make the adaptation immediately, otherwise it can make adaptation following the derived cost-effective time.

Note that in Line 6 of Algorithm 1, we assume that the defender can tell whether he is under being attacked or not. This assumption is realistic against many attacks such as DDoS attacks and covert channel attacks. For example, a defender would suffer from a much higher abnormal traffic volume than usual in a DDoS attack [12]; a defender could perform the detection via side channel analysis to locate a covert channel attack [30]. Our algorithm is classified as a *proactive MTD* approach as it may make an adaptation before a real attack occurs.

Algorithm 2 Decide-adaptation-time procedure DAT ($C_{adp}, C_{att}, Adp_{hist}$)

```
1: Fit a distribution H(t) using Adp_hist;
2: Derive adaptation time T (shown in Equation 1);
3: if Adp_hist does not fit any closed form distribution then
4:     T = mean value of Adp_hist;
5: end if
6: return T
```

Some remarks on Algorithm 2 are given below. Algorithm 2 describes the procedure of deriving the cost-effective adaptation time. The highlight is that we model the problem using a renewal reward process [19] to decide the right timing to make adaptations in order to minimize the overall cost. In the following, we show how to derive Equation 1 in Algorithm 2.

THEOREM 1. *For a family of attacks with their attack time interval lengths satisfying identical and independent distributions, there exists an optimal algorithm to decide the optimal adaptation time with the objective of minimizing the long-run expected cost.*

Note that the family of attacks described in Theorem 1 is broad and it includes covert channel attacks in cloud [18], IP address scanning and stealthy port scanning [10], and password cracking [26], etc. Also, we remark that the attack time interval lengths are values from a continuous real region.

PROOF. The moving target defense process is essentially a stochastic process in which adaptations take place over time. Specifically, the moving target defense process can be modeled as a renewal process that each adaptation can be regarded as a 'restart' of the process that the defender waits for the next adaptation. Thus, we formulate the adaptation timing problem as a renewal process with the objective to decide the 'renew' time (i.e., adaptation time) that is most cost-effective.

We consider the stochastic process in which a defender makes adaptations over time. The cost of making adaptations is characterized as a function of the timing of when to make adaptations. For a defender d, let X_i denote the time length between the $(i-1)$-th and i-th adaptations. Let Y_i denote, after the i-th adaptation of the defender, the time length for the attackers to successfully probe sufficient vulnerability of the defender in order to launch an attack. Assume Y_i has a distribution H_d, where $h(x)$ is the probability density function. The probability that an attacker spends time x to probe sufficient vulnerability is $h(x)$. The cumulative density function is thus $H(x) = \int_0^x xh(x)dx$. We note here that the function $h(x)$ is historical-based.

Let R_i denote the total cost incurred during a time interval with length X_i. Let T_i^d denote the time for a defender to make the i-th adaptation, i.e., when the elapsed time since the $(i-1)$-th adaptation equals to or is larger than T_i^d, the defender will make adaptations. Based on the above definitions, we have

$$X_i = \begin{cases} T_i^d, & \text{if } Y_i > T_i^d \\ Y_i, & \text{if } Y_i < T_i^d \end{cases}$$

The defender will pay an extra cost if it is being attacked, i.e., the attacked cost, denoted as C_{att}^d. Whenever a defender is being attacked, we know that it needs to make an adaptation immediately. The defender pays an adaptation cost when it makes an adaptation, denoted as C_{adp}^d. Thus, we have

$$R_i = \begin{cases} C_{adp}^d, & \text{if } Y_i > T_i^d \\ C_{att}^d + C_{adp}^d, & \text{if } Y_i < T_i^d \end{cases}$$

Let $E(R_i)$ and $E(X_i)$ denote the expected values of R_i and X_i, respectively. We then have

$$\begin{aligned} E(X_i) &= \int_0^{T_i^d} x \cdot h(x)dx + \int_{T_i^d}^{\infty} T_i^d \cdot h(x)dx \\ &= \int_0^{T_i^d} x \cdot h(x)dx + T_i^d \cdot (1 - H(T_i^d)) \\ E(R_i) &= C_{adp}^d \cdot \Pr(Y_i > T_i^d) + (C_{att}^d + C_{adp}^d) \cdot \Pr(Y_i \leq T_i^d) \\ &= C_{adp}^d + C_{att}^d \cdot H(T_i^d) \end{aligned}$$

Let $L(T_i^d)$ denote the cost rate when the time threshold to make adaptations is T_i^d. We have

$$\begin{aligned} L(T_i^d) &= \frac{E(R_i)}{E(X_i)} \\ &= \frac{C_{adp}^d + C_{att}^d \cdot H(T_i^d)}{\int_0^{T_i^d} x \cdot h(x)dx + T_i^d \cdot H^c(T_i^d)} \end{aligned} \quad (1)$$

Recall that our optimization objective is to choose a value for T_i^d to minimize the long-run cost. The T_i^d value which makes the derivative $\frac{\partial L(T_i^d)}{\partial T_i^d} = 0$ minimizes the long-run cost rate $L(T_i^d)$, and thus is the optimal time to make adaptations. □

4. PERFORMANCE EVALUATION

In this section, we evaluate the cost-effectiveness performance of the proposed RRT algorithm against DDoS attacks. (We will case study covert channel attacks in Section 5.) Recall that a defender is able to tell whether a DDoS attack happens or not immediately.

First, we describe the experimental settings. Then we introduce the evaluation methodology and metrics. Finally, we measure the cost efficiency of our proposed RRT algorithm using real attack traces with comparisons to existing strategies, including up-to-date periodical adaptation strategies and a randomized adaptation strategy in [32, 13, 21].

4.1 Experimental settings

The experiments are running on a workstation with 2.30GHz Intel Xeon E5-2630 CPU and 32GB memory. The dataset used in the experiments is introduced in the following.

Dataset.

The dataset used in the simulation is a set of confirmed DDoS attack traces provided by a third party. The traces consist of 50,704 different DDoS attacks on 9,026 different IPs from time period 08/29/2012 to 03/24/2013 [24]. The dataset contains the information of each attack, including *attacker id*, *victim (defender) IP* and *attack timestamp*.

Attack traces.

An attack trace contains all the timestamps along the timeline that a defender is being attacked by an attacker which is a bot family. In the simulation, we simulate the arrivals of attacks on each defender using its corresponding attack traces in the dataset. As the dataset is collected in a relatively short time period, some defenders are only observed to be attacked a few times. Since our objective is to characterize the long-term cost efficiency of different moving target algorithms, we use the traces with attack counts larger than 30 as the input of our simulation. There are in total 141 such traces.

Attack intervals distributions.

An attack (time) interval is the time difference between two successive attacks. The distribution of the attack intervals is critical to the RRT algorithm since it takes the attack interval history as input and makes decisions about the right timing to make adaptations based on the attack interval distributions. In case the historical attack intervals satisfy none of the statistic distributions, we use the mean value of the historical intervals as the adaptation time in the RRT algorithm.

We fit the 144 attack traces against various well-known probability distributions including LOG-NORMAL, EXPONENTIAL, UNIFORM, NORMAL and POISSON DISTRIBUTION. We use Kolmogorov-Smirnov [5] statistics to test the *goodness-of-fit* with a significance level setting as 0.05. We find 98 out of 144, i.e., around 70%, of the attack traces have their attack intervals following a LOG-NORMAL distribution. While 39 of the traces fit none of the distributions tested. If we tune the significance level setting to a larger number, then

we will have more attack traces satisfying LOG-NORMAL distributions.

Figure 3 shows the attack trace distribution based on each victim. In the figure, the x-axis corresponds to each victim, while the y-axis shows the number of attackers that have attacked each victim as well as the distribution of the attack traces. Different colors are used to show different attack trace distributions.

Figure 4 further plots the attack traces distributions based on each attacker. In particular, in Figure 4, the x-axis shows the attacker ID with obscuration, while the y-axis shows the total count of victims (distinguished by IP address) each attacker has attacked as well as the distribution of each attack trace. Different attack trace distributions are shown using different colors. From the figure, we can see that for an attacker, its attack distributions on different victims may follow the same distribution, e.g., attacker *25B93033-3** and *089AF274-2** both have LOG-NORMAL distribution attack time intervals on all their victims.

These figures confirm our condition given in Theorem 1 that the attack time intervals satisfy an identical and independent distribution.

Attacked cost.

The attacked cost is defined as the cost that a defender suffers from an attack. According to the study commissioned by Incapsula [14], the average cost of DDoS attacks is $\$40,000$ per hour. Most of the organizations have the attacked costs in the range from $\$5,000$ to $\$100,000$.

We simulate the attacked cost of all the defenders following the detailed attacked cost distributions in [14] with the recovering time from failure being set as 5 minutes. We acknowledge that more specific analysis will be required to decide the attacked cost of different organizations in terms of different security threats. For example, for many medium and large organizations, they often have their own security department in performing the risk analysis and measuring the potential attacked cost. While for small organizations or individuals, some tools have been developed and made available for them to measure the attacked cost [1]. Measuring the attacked cost in terms of various potential attack scenarios is an interesting research topic, which, however, is out the scope of our paper.

Adaptation cost.

The adaptation cost refers to the impact of adaptations on system performance and service quality. For example, in terms of DDoS attacks, a moving target defense mechanism changes the IP addresses of the system dynamically [3]. The overhead of making adaptations translates to the time overhead of updating the routing and firewall configuration after each adaptation [3]. According to the study in [10], the latency of moving target defense mechanisms grows linearly as the number of packets to be processed increases. For example, for processing $20,000$ packets, the time overhead is 300ms. Thus the latency varies among different organizations as they might have different traffic rates. We translate the latency into the business cost according to the study in [7] based on the average cost rate of media organizations surveyed. For small (latency 200ms), medium (latency 2s), large size (latency 20s) organizations, the cost of making an adaptation is \$1, \$10 and \$100, respectively.

Deterrence ratio.

Deterrence ratio has been used to characterize the effect of moving target defense [10]. As in the moving target defense mechanism, defenders dynamically change the network or system configurations to invalidate the information gained by attackers, the attackers are forced to frequently redo the reconnaissance activities in order to regain the lost information to launch an attack. Thus moving target defense has the effects of delaying the arrival of attacks [10]. The *deterrence ratio* is used to measure the delaying factors, which is defined as follows

$$deterrence\ ratio = \frac{attack\ time\ interval\ with\ MTD}{attack\ time\ interval\ without\ MTD}$$

Due to the limitation of the dataset, we are not aware whether the defenders in the dataset have employed moving target defense mechanisms or not. This issue matters as our proposed algorithm is dedicated for defenders to use under the moving target defense scheme. In essence, our designed algorithm aims to assist in making decisions about the right timing to make adaptations. Thus we consider the following two cases.

- *Case 1:* In the collected dataset, the defenders did make adaptations.

  In this case, as the attack traces capture the effect of moving target defense, we use the original attack traces to simulate the attacks in the simulation.

- *Case 2:* In the collected dataset, the defenders did not make adaptations.

  In this case, we need to incorporate the effects of moving target defense into the attack traces. Specifically, if the defenders make adaptations to defend against DDoS attacks, e.g., changing the firewall filtering rules or changing IP addresses [3], it will have the effects of delaying next attack as the attackers will need more time to exploit the vulnerability of defenders (e.g., IP scanning, training smart bots to avoid being identified by filtering rules) in order to launch the attack. Thus, we increase the time between each two successive attacks in the attack traces to reflect the effects of moving target defense. In the simulation, we study the cost efficiency under the attack traces with various deterrence ratios.

4.2 Evaluation Methodology

In the following, we first overview the simulation procedure, and then introduce the algorithms used for comparisons as well as the measurement metric.

Simulation overview.

We simulate the moving target defense process independently for each defender. The attack scenario is characterized by the attack traces from the real attack dataset. Each defender makes adaptations according to the adaptation time decision made by the algorithms. Each algorithm maintains the attacked timestamp histories, while the future attack timestamps are unknown to all the algorithms. The simulation for each defender continues until the end of the attack traces.

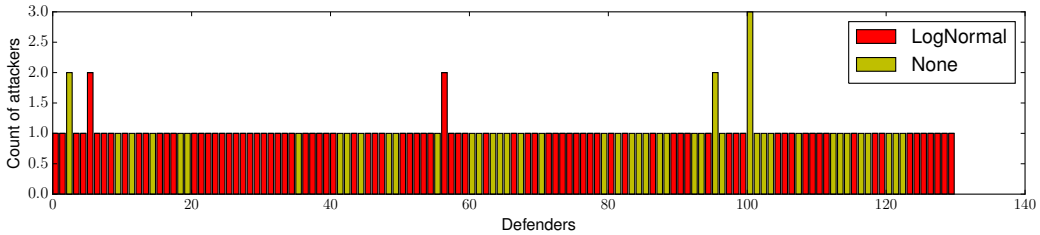


Figure 3: Attack interval distribution per defender. The y-axis is the count of attackers that have attacked each defender.

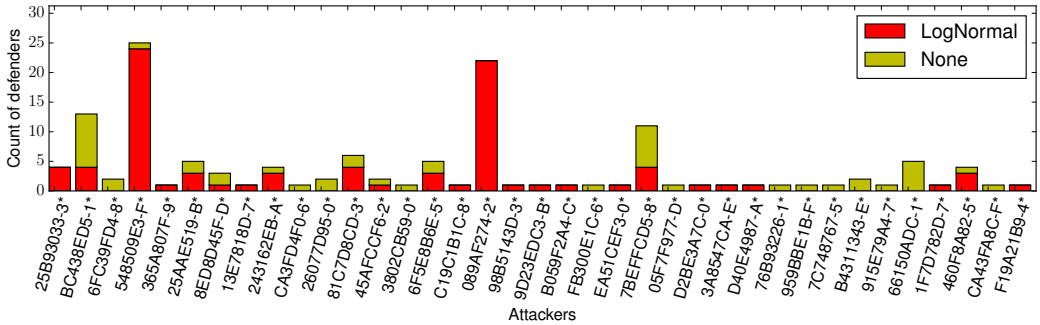


Figure 4: Attack interval distribution per attacker. The y-axis is the count of victims (defenders) each attacker has attacked.

Algorithms for comparisons.

In the simulation, we compare our proposed algorithm RRT against the up-to-date periodically adaptation algorithms adopted in [32, 13, 21]. We also include a randomized algorithm for comparison. All the algorithms are described as follows.

ALG_{mean}, ALG_{low}, ALG_{median}, ALG_{high}, ALG_{higher}: These algorithms make adaptations when the elapsed time since last adaptation exceeds the mean value, 25th, 50th, 75th, 100th percentile of the historical attack time intervals, respectively, or when under attacks.

ALG_{random}: The algorithm makes adaptations when the elapsed time since last adaptation exceeds a value which is randomly selected from the historical attack time intervals, or when under attacks.

Long-term cost rate.

For each defender, we record the adaptation cost and the attacked cost it suffers under different adaptation algorithms separately in the simulation. We also record the elapsed time for each defender during the simulation. Then the *long-term cost rate* of each defender is defined as $\frac{\text{total cost}}{\text{total elapsed time}}$, where *total cost* is the sum of the *adaptation cost* and the *attacked cost*. In comparing the performance of different adaptation algorithms, we compare the average *long-term cost rate* of all defenders under each algorithm.

4.3 Evaluation Results

We first study the long-term cost rate of all the defenders under various algorithms. Then we investigate the cost rate under various deterrence ratios using the proposed RRT algorithm as an example. Finally, we study the adaptation time of the algorithms under various adaptation cost settings and we discuss the running time of the algorithms.

Evaluation of cost efficiency.

We simulate different algorithms under various settings of adaptation costs, and compare the average long-term cost rates of all the defenders.

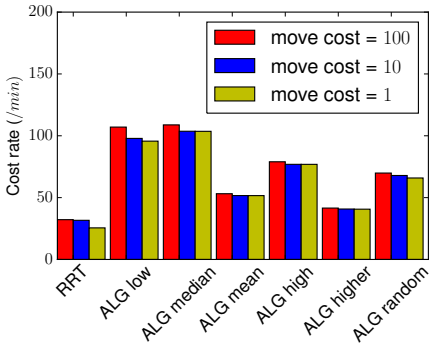
Figure 5 shows the long-term cost rate of all defenders under various algorithms, with the adaptation costs set as \$100, \$10 and \$1. In particular, Figure 5(a) uses the original attack traces as the attack traces, while Figure 5(b) uses the original attack traces with deterrence ratio set as 2.

From Figure 5, we can see that ALG_{low} has a very high cost since it makes adaptations frequently without taking into account the attackers' power and the adaptation costs. Besides, its cost rate tends to grow faster than the other algorithms when its adaptation cost increases from \$1 to \$100. As it makes adaptation very frequently in order to avoid being attacked, its long-term cost rate is more sensitive to the adaptation cost.

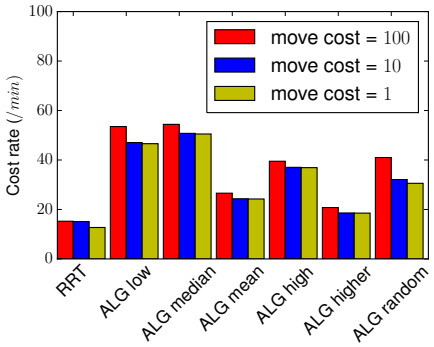
In contrast, ALG_{higher} tends to be less sensitive when the adaptation cost grows. But it has a higher cost rate compared with RRT especially when the adaptation cost is relatively low (e.g., \$1 per adaptation). The reason is because ALG_{higher} always makes adaptation with adaptation time interval equals to the maximum value of the historical attack intervals, thus it is always very likely to be attacked.

ALG_{random} has a cost rate much higher than RRT and ALG_{higher} as shown in Figure 5. The underlying reason is because randomly determining the time to make adaptations might introduce more adaptation cost if the adaptations are made frequently, or it might incur a higher cost of being attacked if adaptations are not made frequently enough. This indicates a random algorithm is not capable of trading off the adaptation cost and the cost of being attacked if the attack history is not involved in the randomness.

We further investigate the long-term cost rate of each single defender and again we find similar results, related figures are omitted here. We conclude that unlike the fixed-



(a) Original attack traces



(b) Attack traces with deterrence ratio = 2

Figure 5: Average cost rate of all defenders under various adaptation algorithms. The adaptation cost (i.e., move cost) is set as $100, $10, and $1 respectively.

periodic or randomized algorithms, **RRT considers the adaptation cost and the attacked cost, and it makes a trade-off in between. Thus, it tends to have a better cost rate than other algorithms.** Above all, we conclude **RRT is more economical/cost-effective compared with the fixed-periodical adaptation and randomized adaptation algorithms.**

Deterrence ratio.

Comparing Figure 5(a) and Figure 5(b), we find that through modifying the attack traces with setting deterrence ratio as 2, the cost rate is almost reduced by half. The reason is because with a deterrence ratio set as 2, the attack time intervals are increased by one time, while the adaptation cost and attacked cost remain the same. However, the cost that a defender suffers might be affected as the distribution-fitting under the modified attack traces might be different, and thus might affect the adaptation time decision making for RRT. Motivated by this, we further explore the relationship between the long-term cost rate and the deterrence ratio.

We conduct simulations to study the long-term cost rate under various deterrence ratios using our proposed RRT algorithm as an example. The study results are shown in Figure 6.

Figure 6 demonstrates cost rate under RRT algorithm with the adaptation cost rate set as $100. In the figure, the

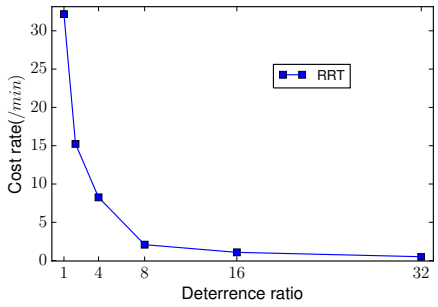


Figure 6: Cost rate vs. deterrence ratio under RRT algorithm with the adaptation cost being set as $100.

.

x-axis shows the deterrence ratio, while the y-axis shows the corresponding long-term cost rate. From the figure, we can see that the long-term cost rate decreases at almost the same speed as the deterrence ratio increases. This result indicates if the defender can find a moving target defense mechanism to delay the attack as much as possible, it can reduce the long-term cost rate accordingly.

Adaptation interval.

In the following, we study the adaptation time of different algorithms under various adaptation cost settings. We conduct simulations in which defenders making decisions based on each algorithm. We record the adaptation time under each algorithm and compute the average value in Figure 7.

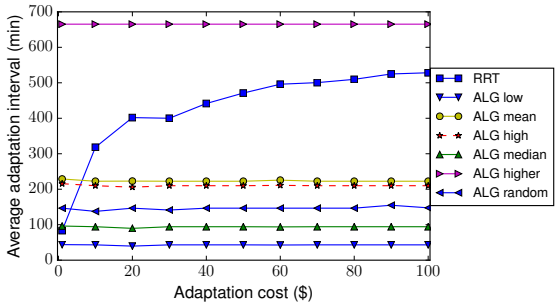


Figure 7: Average adaptation time. Deterrence ratio = 1.

.

In Figure 7, the x-axis shows the adaptation cost range from $1 to $100, and the y-axis shows the corresponding average adaptation time interval under each algorithm. From the figure, we can see that as the adaptation cost increases, the RRT algorithm tends to have a larger average adaptation time interval, i.e., lower adaptation frequencies. This indicates that when the adaptation cost is high, it is no longer economical for making adaptations frequently. While for the other algorithms, as they make adaptation time decisions without considering the cost, the average adaptation time under these algorithms is not affected by the adaptation cost.

Running time.

We implement the proposed RRT algorithm using Mathematica [28]. Its average running time is 1.74s which is dom-

inated mainly by solving Equation 1. The running time is negligible compared with the adaptation time interval which is in the scale of hundreds of minutes as shown in Figure 7. We remark that further optimization can be done by maintaining a dynamic table to keep track of the optimal adaptation time in Equation 1. Therefore, the running time can be reduced to milliseconds if there is a match of the optimization problem in the table. Besides, for a defense system which has multiple servers that require separate adaptations, the RRT algorithm can be easily scaled up to run in parallel, as each optimization is independent from each other.

5. A CASE STUDY OF COVERT CHANNEL ATTACKS

In this section, we further conduct a case study to investigate the cost rates of employing the moving target defense with various adaptation strategies to combat covert channel attacks in the cloud.

In this case study, we conduct simulations to evaluate the average cost of the defenders in defending against covert channels using virtual machine migration mechanism [15]. Specifically, we build a cloud testbed in which defenders can migrate the Virtual Machines (VMs) from one physical host to another in order to defend against covert channel attacks. A defender will suffer a cost if being attacked as described in Section 4.1. We introduce the settings of adaptation costs (i.e., the costs of making migrations) and the attack traces in the following.

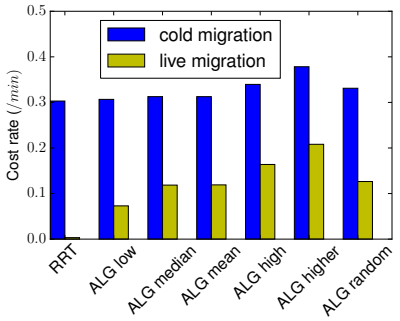
For the adaptation costs, we consider the impact of latency due to VM migrations and translate the latency into business costs [7]. We conduct experiments to measure the VM migration latency. There are mainly two VM migration techniques, i.e., *cold migration* and *live (hot) migration*. Cold migration stops the VM in its current host, copies it to a remote host and then restarts it. Cold migration incurs high service latency since the service is shut down during the whole migration process and the migration could be time-consuming if the disk volume is large. While live migration keeps the VM running in its current host and copies the VM to remote hosts, and then suspends the VM for making the final copy of dirty pages to remote hosts, and finally continues running the VM in the new host. Thus live migration causes less service latency compared with cold migration.

We measure VM migration latency in two popular cloud building tools including OpenStack [2] and VMware vSphere [9]. We use three machines in building the experiment testbed. The machines have 16G, 16G, and 12G RAM, respectively. All three machines have 4 core processors. Each two of them are connected by 1G Ethernet LAN.

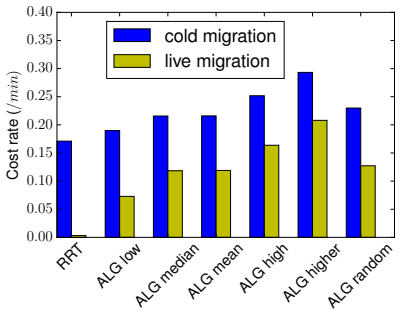
We measure VM migration latency under various VM settings, including different RAM sizes and disk sizes. The live migration latencies under OpenStack and VMware are in average 0.806s and 0.61s, respectively, which are translated to the monetary costs [7] as $4.3 and $3.05, respectively. While in both testbeds, the cold migration latency grows almost linearly as the used storage increased. The cold migration latency for a VM with 4G RAM and 20G allocated disks (with 10G used) for OpenStack and VMware are in average 545s and 273s, respectively, translating to business costs as $2725 and $1366, respectively.

For the cloud covert channel attack trace, we simulate it using a uniform distribution with statistics from the latest cloud covert channel attack study [29]. The attack time interval is in the range of 50 to 150 minutes with deterrence ratio set as 160 [10]. The deterrence ratio will keep increasing as the size of the cloud gets larger. Since a larger cloud provides more physical hosts for the VMs to migrate to and thus increases the uncertainty for the attackers to pinpoint the target.

The cost rate under various ways of virtual migration is shown in Figure 8. The cost rate not only includes the cost of making adaptations (i.e., migration), but also includes the cost of being attacked if adaptations are not conducted frequently enough to avoid the attacks.



(a) OpenStack



(b) VMware vSphere

Figure 8: Long-term cost rate for moving target defense with virtual machine migration under OpenStack and VMware vSphere Cloud infrastructure. Note for live migration, RRT has cost rate $0.0032/min and $0.0031/min in (a) and (b), respectively.

From Figure 8, we can see with live migration, the average cost of using RRT to defend co-residence attacks is around $0.003/min. While, the latest cloud covert channel study [29] demonstrates that it costs attackers around 100 minutes in achieving co-resident with the target defender with a cost about $32. That means that the cost of the defender is lower compared to the cost that paid by the attackers ($0.32/min). In specific, with live migration, the attackers pay around 106 times of what paid by the defenders. It implies that **cloud providers can offer such a protection mechanism as a value-added security service to combat covert channel attacks so as to protect the defenders while discouraging potential attackers**.

While for cold migration, defenders need to pay a much higher cost compared with live migration. In this case, the cost rate under OpenStack and VMware vSphere cloud in-

frastructure are \$0.17/min and \$0.30/min, respectively under our proposed algorithm. Compared with live migration, cold migration has a much higher cost rate since it incurs much longer service downtime during VM migration. This result indicates that live migration is preferable in fighting covert channel attacks with MTD schemes. Overall, the results in Figure 8 also indicates the importance of smart adaptations. In specific, with a live migration scheme, as the migration cost is almost negligible, a frequent adaptation strategy outperforms the others, while with a cold migration scheme, as the migration cost is high, a frequent adaptation brings lots of costs. Under whatever cases, our proposed RRT algorithm can balance the cost of migration and the cost of being attacked, and thus achieve lower cost compared with other periodic or randomized algorithms.

We further consider the cost rate under the static VM placement mechanism adopted in current clouds [29]. As the attackers can find the static VM in around 100 minutes, and the cost of being attacked is at least be \$5000 per hour [14]. Even if we are considering the optimistic case in which the service can be restored in 5 minutes, the cost rate of the defender is still about \$4.16/min. The high cost of defenders allures the attackers to conduct attacks to achieve their malicious purpose. Compared with the static VM placement algorithm, **a moving target defense mechanism with our proposed algorithm has a much lower cost rate. This indicates moving target defense is a cost-effective mechanism to defense against covert channel attacks in a cloud environment**.

6. RELATED WORK

In current defense systems, a defender often relies on a static system configuration which makes it easy for attackers to exploit the vulnerabilities. Moving target defense is proposed to eliminate the asymmetric disadvantages between defenders and attackers by making the defense system dynamic [17]. In specific, MTD works by continuously changing the static nature of the network topology, system configurations, execution environment etc. of a defense system and thus invalidate the privileges gained by the attackers and hinder the effectiveness of attacks.

Current research on MTD mainly focuses on designing defense mechanisms for various aspects of the protected system. The designed techniques can be further divided into two categories: *reactive MTD* [25, 11] and *proactive MTD* [21, 8, 13, 4]. For reactive MTD, the defenders make adaptations in a lazy manner only when they are under attacks. While for proactive MTD, the defenders also make adaptations before they are attacked, aiming to thwart attacks proactively and to reduce the negative impact of attacks.

Among the works of proactive MTD, various defense mechanisms are proposed to change different aspects of the defense system configurations such as network addresses, port numbers, operating systems, execution environments etc. For example, Thompson *et al.* [21] conduct experiments to study the effectiveness of moving target defense mechanisms by rotating the operating system in every fixed 60 seconds. Gillani et al. [8] propose a moving target defense mechanism against DDoS attacks by periodically changing the footprint of critical resources. Li *et al.* [13] develop an approach to periodically migrate a virtual machine from one physical host to the other in order to improve the virtual machine survivability in the cloud.

The main challenge in designing proactive MTD mechanisms is to decide the right timing of making adaptations. Most of the existing MTD mechanisms employ either fixed-periodically adaptation strategy or randomized adaptation mechanisms [8, 13, 21]. However, as the adaptation could be costly, the problem of when is the right timing to make adaptation is critical in designing economical proactive MTD. Deloach *et al.* [6] provide techniques to decide the best timing of making adaptations based on vulnerability scanning and intrusion detection tools. However the reliance on intrusion detection to trigger alerts for making adaptation limits the use of this mechanism. As the attackers become smarter and more powerful, the intrusion detection techniques may fail to detect the attacks in a timely and accurately manner [20, 10].

7. CONCLUSION

Moving target defense mechanisms are emerging as an effective approach in cyber-security. It works by changing a defender system's attack surface dynamically and thus increases the difficulty of successful attacks. In this paper, we investigate the problem of when is the right timing to make adaptations in moving target defense in order to be cost-effective in a long-term process. We propose an algorithmic approach based on renewal reward theory. Our algorithm makes an optimal balance between the cost of making adaptations and the cost of being attacked. By evaluating the algorithms using real DDoS and covert channel attack traces, we show that our proposed algorithm is more cost-effective under the same security guarantees, compared with the existing periodical and randomized adaptation algorithms.

Acknowledgments

We appreciate constructive comments from anonymous referees. This work is partially supported by an ARO grant W911NF-15-1-0262 and a NSF grant CNS-1524462.

8. REFERENCES

[1] Calculate your DDoS attack costs. https://www.neustar.biz/resources/tools/ddos-attack-cost-calculator.

[2] Openstack. https://www.openstack.org/, 2015.

[3] Ehab Al-Shaer, Qi Duan, and JafarHaadi Jafarian. Random host mutation for moving target defense. In *Security and Privacy in Communication Networks*, volume 106 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 310–327. 2013.

[4] M. Carvalho and R. Ford. Moving-target defenses for computer networks. *IEEE Security & Privacy*, 12(2):73–76, 2014.

[5] Aaron Clauset, Cosma Rohilla Shalizi, and M. E. J. Newman. Power-law distributions in empirical data. *SIAM Rev.*, 51(4):661–703, 2009.

[6] ScottA. DeLoach, Xinming Ou, Rui Zhuang, and Su Zhang. Model-driven, moving-target defense for enterprise network security. *Models@run.time*, 8378:137–161, 2014.

[7] ECESSA. What does network downtime & latency cost your business? http://www.ecessa.com/blog/downtime-costs/, 2014.

[8] F. Gillani, E. Al-Shaer, S. Lo, Qi Duan, M. Ammar, and E. Zegura. Agile virtualized infrastructure to proactively defend against cyber attacks. In *Proceedings of the 34th IEEE Conference on Computer Communications (INFOCOM)*, pages 729–737, 2015.

[9] VMware Inc. Vmware vsphere 6. https://www.vmware.com/products/vsphere, 2015.

[10] J.H. Jafarian, E. Al-Shaer, and Qi Duan. Adversary-aware ip address randomization for proactive agility against sophisticated attackers. In *Proceedings of the 34th IEEE Conference on Computer Communications (INFOCOM)*, pages 738–746, 2015.

[11] Quan Jia, Huangxin Wang, D. Fleck, Fei Li, A. Stavrou, and W. Powell. Catch me if you can: A cloud-enabled DDoS defense. In *Proceedings of the 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 264–275, 2014.

[12] Shuyuan Jin and D. S. Yeung. A covariance analysis model for ddos attack detection. In *Proceedings of the 2004 IEEE International Conference on Communications*, volume 4, pages 1882–1886, 2004.

[13] Min Li, Yulong Zhang, Kun Bai, Wanyu Zang, Meng Yu, and Xubin He. Improving cloud survivability through dependency based virtual machine placement. In *Proceedings of the International Conference on Security and Cryptograph (SECRYPT)*, pages 321–326, 2012.

[14] Tim Matthews. Incapsula survey : What DDoS attacks really cost businesses. http://lp.incapsula.com/rs/incapsulainc/images/eBook%20-%20DDoS%20Impact%20Survey.pdf, 2014.

[15] Soo-Jin Moon, Vyas Sekar, and Michael K. Reiter. Nomad: Mitigating arbitrary cloud side channels via provider-assisted migration. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 1595–1606, 2015.

[16] Department of Homeland Security. Moving target defense. http://www.dhs.gov/science-and-technology/csd-mtd, 2011.

[17] P. Pal, R. Schantz, A. Paulos, and B. Benyo. Managed execution environment as a moving-target defense infrastructure. *IEEE Security Privacy*, 12(2):51–59, 2014.

[18] Thomas Ristenpart, Eran Tromer, Hovav Shacham, and Stefan Savage. Hey, you, get off of my cloud: Exploring information leakage in third-party compute clouds. In *Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS)*, pages 199–212, 2009.

[19] Sheldon M. Ross. *Introduction to Probability Models*, 1972.

[20] Shad Stafford and Jun Li. Behavior-based worm detectors compared. In *Recent Advances in Intrusion Detection*, volume 6307 of *Lecture Notes in Computer Science*, pages 38–57. 2010.

[21] M. Thompson, N. Evans, and V. Kisekka. Multiple os rotational environment an implemented moving target defense. In *In proceedings of the 7th International Symposium on Resilient Control Systems (ISRCS)*, pages 1–6, 2014.

[22] C. Tunc, F. Fargo, Y. Al-Nashif, S. Hariri, and J. Hughes. Autonomic resilient cloud management design and evaluation (arcm). In *Proceedings of the International Conference on Cloud and Autonomic Computing (ICCAC)*, pages 44–49, 2014.

[23] Venkatanathan Varadarajan, Yinqian Zhang, Thomas Ristenpart, and Michael Swift. A placement vulnerability study in multi-tenant public clouds. In *Proceedings of the 24th USENIX Security Symposium (USENIX Security)*, pages 913–928, 2015.

[24] An Wang, Wentao Chang, Aziz Mohaisen, and Songqing Chen. Delving into internet DDoS attacks by botnets: characterization and analysis. In *Proceedings of the 45th IEEE International Conference on Dependable Systems and Networks (DSN)*, 2015.

[25] Huangxin Wang, Quan Jia, Dan Fleck, Walter Powell, Fei Li, and Angelos Stavrou. A moving target DDoS defense mechanism. *Computer Communications*, 46:10 – 21, 2014.

[26] M. Weir, S. Aggarwal, B. de Medeiros, and B. Glodek. Password cracking using probabilistic context-free grammars. In *Proceedings of the 30th IEEE Symposium on Security and Privacy*, pages 391–405, 2009.

[27] Michael L. Winterrose, Kevin M. Carter, N. Wagner, and William W. Streilein. Adaptive attacker strategy development against moving target cyber defenses. *CoRR*, abs/1407.8540, 2014.

[28] Wolfram Research, Inc. Mathematica 8.0.

[29] Zhang Xu, Haining Wang, and Zhenyu Wu. A measurement study on co-residence threat inside the cloud. In *Proceedings of the 24th USENIX Security Symposium (USENIX Security)*, pages 929–944, 2015.

[30] Yinqian Zhang, Ari Juels, Alina Oprea, and Michael K. Reiter. Homealone: Co-residency detection in the cloud via side-channel analysis. In *Proceedings of the 32nd IEEE Symposium on Security and Privacy*, pages 313–328, 2011.

[31] Rui Zhuang, Alexandru G. Bardas, Scott A. DeLoach, and Xinming Ou. A theory of cyber attacks a step towards analyzing mtd systems. In *Proceedings of the 2nd ACM Workshop on Moving Target Defense (MTD)*, 2015.

[32] Rui Zhuang, Scott A. DeLoach, and Xinming Ou. A model for analyzing the effect of moving target defenses on enterprise networks. In *Proceedings of the 9th Annual Cyber and Information Security Research Conference (CISR)*, pages 73–76, 2014.