# Model-based evaluation of combinations of Shuffle and Diversity MTD techniques on the cloud

Hooman Alavizadeh, Dong Seong Kim, Julian Jang-Jaccard

PII: S0167-739X(19)31518-3
DOI: https://doi.org/10.1016/j.future.2019.10.009
Reference: FUTURE 5234

To appear in: *Future Generation Computer Systems*

Received date : 11 June 2019
Revised date : 25 August 2019
Accepted date : 18 October 2019

- Providing formal mathematical definitions for the combination of Shuffle (S) and Diversity (D) MTD techniques.
- Proposing new mechanisms to combine Shuffle (S) and Diversity (D) techniques.
- Quantify and comparing the cloud security posture using Graphical Security Models (GSMs).
- Deploying MTD techniques using simulation and evaluating the MTD techniques using four security metrics: Attack Cost, Cloud Risk, Return on Attack, and Attack Success Probability.

# Model-based Evaluation of Combinations of Shuffle and Diversity MTD Techniques on the Cloud

Hooman Alavizadeh[a], Dong Seong Kim[b], Julian Jang-Jaccard[a]

[a]*School of Natural and Computational Sciences, Massey University, Auckland, New Zealand*
[b]*School of Information Technology and Electrical Engineering, University of Queensland, Australia*

## Abstract

Regardless of cloud computing capabilities, security is still one of the biggest threats in the cloud. Moving Target Defense (MTD) has shown to be an effective security mechanism to secure the cloud by changing the attack surface to make uncertainties for the attackers. In this paper, we propose a combination of two MTD techniques: Shuffle and Diversity which we believe further attributes to reduce the cyber attack surface. We first provide the formal definitions of the combination to design and implement our proposal. Then, we investigate a number of approaches in which Shuffle and Diversity can be combined in order to provide the most effective defense. Towards, we utilize Network Centrality Measures (NCMs) to find out the most critical component in the cloud. Then, we evaluate the proposed MTD techniques through formal Graphical Security Models (GSM) and quantify the cloud security level through security metrics before and after deploying the MTD techniques. Our experimental evaluation shows that the combination of Shuffle and Diversity techniques can increase the security posture of the cloud.

*Keywords:* Security analysis, Moving Target Defense, Cloud Computing, Security Metrics

*Email addresses:* `h.alavizadeh@massey.ac.nz` (Hooman Alavizadeh), `dan.kim@uq.edu.au` (Dong Seong Kim), `j.jang-jaccard@massey.ac.nz` (Julian Jang-Jaccard)

## 1. Introduction

Cloud computing security has become a huge challenge for the cloud providers as the cloud's customers cannot trust the security of this new paradigm while the cloud provides comprehensive services to their customers. According to the International Data Corporation (IDC) survey on the cloud computing challenges, the cloud security with 87.5% was ranked first as the greatest concern for the enterprise cloud customers [1]. Conventional security mechanisms are used to address the security issues by eliminating the vulnerabilities and risks. However, it is difficult to perfectly remove or patch all possible vulnerabilities on a system. Hence, it is crucial to have effective security mechanism to improve the cloud security from different defensive aspects [2, 3]. As an emerging proactive approach, Moving Target Defense (MTD) has been proposed which can provide another perspective of defensive strategies against cyber attacks. MTD makes a system more unpredictable for the attackers by continuously changing the attack surface. MTD can utilize the existing system components and technologies providing more affordable defense solutions.

In [4], Hong *et al.* categorized MTD techniques into three comprehensive categories including [4]: Shuffle [5, 6], Redundancy [7] and Diversity [8, 9]. In general, Shuffle MTD techniques can reconfigure the system's components in order to change the attack surface and consequently increase uncertainty and confusion for the attacker. Redundancy MTD techniques deal with replication of any system's component aiming to enhance the system and service reliability or availability for the customers. Thus, if a system's component fails due to attack, there would be alternative ways to provide the same service. The advent of the Internet of Things (IoT) makes more viable attack sources for attackers so that they can launch various attacks through the botnets these days. Botnets are good starting points for attackers to launch a wider attack range to the cloud using vulnerable IoT-based botnets [10]. For example, a Distributed Denial-of-Service (DDoS) attack utilizes botnets (leveraging compromised IoT devices) to attack a cloud by flooding traffic messages from various sources aiming to deny services to users. In this case, Redundancy contributes to increase cloud resiliency, and can battle these kinds of attacks. However, the investigation of Redundancy technique is out of the scope of this paper and is presented in [11]. Diversity MTD techniques may increase the difficulties of attacks by changing the system's component variant. Changing a component in the system may introduce different and new set of vulnerabilities and invalidate the vulnerability information collected by the attackers. Ultimately, the attacker may spend more time, effort, and money to learn new techniques to exploit the newly introduced vulnerabilities.

Deploying an MTD technique for a specific reason may vary the security posture of a system. Most of proposed MTD techniques do not offer convincing evidence if they would be effective as claimed. Therefore, it is important to assess the effectiveness of MTD techniques through security metrics such as Attack Cost (AC) and Return on Attack (RoA) which evaluate the security from the attackers' perspective and other metrics like Risk (R) and Attack Success Probability (ASP) which may be desirable metrics for cloud providers' perspective. Security analysis plays an inevitable role in evaluating the overall security-related perspectives of a system.

Formal graphical attack models like Graphical Security Models (GSMs) are useful tools to model and evaluate the security of the systems such as IoT and enterprises [12] or clouds [13]. GSMs can be used to evaluate the effectiveness of MTD techniques [11, 4]. However, analyzing the security through most of GSM suffers from exponential computational complexity issue, especially, in the large networks [14]. To overcome this problem, Hierarchical Attack Representation Model (HARM) is proposed which is a formal hierarchical graph-based model including two layers [14]. HARM is more scalable and adaptable than other formal GSMs [15]. In this paper, we use HARM to evaluate the effectiveness of the MTD techniques and compute the security metrics.

MTD techniques can be either used independently or combined together to obtain more effective results. Many MTD strategies have been proposed [16, 17], but it is still difficult to evaluate the effectiveness of combined MTD techniques.

Combining MTD techniques can introduce additional benefits of enhancing security, which may not be possible under a single technique based MTD solutions; for instance, as Redundancy is mostly used to increase service reliability, it can be measured with the concepts of system dependability (e.g. reliability), while other MTD techniques like Shuffle and Diversity are used to increase the security of a system and need to be evaluated using security metrics. Thus, MTD techniques can be well-mingled together aiming to increase both security and reliability. However, those techniques should be evaluated using adequate security metrics as deploying each MTD technique may affect others in different ways. A combination of MTD techniques including Shuffle, Diversity, and Redundancy is presented in [18] which is mainly limited to a single deployment strategy and four security metrics such as R, AC, RoA, and Reliability. However, in this paper, we conduct extensive analysis on Shuffle and Diversity MTD technique which considers different sets of combination strategies. Moreover, we capture the effects of different MTD deployment strategies using eight important security metrics.

The earlier version of this paper was presented in [19]. In this paper, we extend the earlier version mainly focusing on formalism and definitions of MTD techniques and combination strategies on the cloud together with more effective security metrics to show the different perspective of the cloud security posture affected by deploying MTD techniques. Moreover, we have revised the previous model with new vulnerabilities and metrics. The new contributions of this paper, which to the best of our knowledge have not already been proposed by other works, are listed as follows:

- We provide the formal mathematical definitions for the combination of Shuffle (S) and Diversity (D) MTD techniques to unambiguously design and implement it in the cloud. Our formal method is written based on Hierarchical Attack Representation Model (HARM).
- We propose a new approach that combines Shuffle (S) and Diversity (D) techniques. We also provide a set of strategies for the way Shuffle (S) and

Diversity (D) can be combined differently. By computing Important Measures (IMs), the effects of the different combination strategies are calculated and explained.

- We provide simulation and calculation results for the deployed MTD techniques using four security metrics to assist in extensive understanding of the trades between MTD techniques and Metrics involved in the combined defense and the attack. The security metrics we use include: Cloud Risk (R), Attack Cost (AC), Return on Attack (RoA), and Attack Success Probability (ASP).
- We also include the path-based security metrics and evaluate them against each MTD strategy to evaluate how difficult the attacker can reach a target. We also conduct regression analysis between path-based metrics and security metrics (R and ASP) to investigate the correlation between those metrics.
- We perform comparative analysis and evaluation of "before" and "after" deployment of MTD techniques to be able to quantify and compare the cloud security posture.

The rest of this paper is organized as follows. We present the related work in Section 2. We define the concepts, definitions, formalism, and the security metrics used throughout the paper in Section 3. In Section 4, we provide definitions and formalism for MTD techniques including Shuffle and Diversity and evaluate the deployment of each MTD technique. Definition, deployment, and analysis of different strategies for combining MTD techniques are given in Section 5. Then further discussion and limitations are given in Section 6. Finally, we conclude the paper in Section 7.

## 2. Related Work

Definition of MTD is not restricted to a portion or a specific part of a system. Any static or dynamic component of a system can be changed using MTD techniques to make a system more unpredictable for attackers. MTD can be deployed through different layers of a network. Numerous researches have been proposed either to introduce new techniques or improve an MTD model [20, 5].

Many researchers have focused on MTD frameworks [21, 22], applications [23, 24], strategies and techniques [5, 25, 26]. However, we summarized the proposed MTD techniques based on different contexts.

Redundancy can be deployed on the cloud using replication of the cloud resources (such as VM replication). This technique can increase the reliability of the cloud by creating redundant VMs (e.g. crucial servers). The extension of cloud resources to increase the service reliability and battle against DDoS attacks have been investigated in [11, 7]. However, deploying redundant cloud resources to avoid attacks may affect cloud's performance. To address this problem, Al-Haidari *et al.* [27] studied the impact of cloud scaling size factors against the CPU cost and performance. They used the optimization problem to find an optimal number of cloud resources against QoS requirements. In [28], the authors proposed a resource allocation scheme for the virtual desktop cloud (VDC) and developed a tool named VDC-Analyst aiming to satisfy quality of experiments (QoE) for users by increasing net utility and service response time.

Jafarian *et al.* [5] developed an MTD technique to proactively change the IP addresses of the hosts. Similarly, the concept of Random Route Mutation (RRM) has been introduced by Al-Shaer [25] to find an optimal randomized path between source and target. Antonatos *et al.* [29] introduced a Shuffle (i.e., IP randomization technique) MTD technique to thwart Hit-List worms attacks. The implemented IP shuffle method avoids malicious worm to gather information about the victims (i.e., target hosts) in a networked system, which made it harder for the worm to identify new targets. They evaluated their method by assessing the address changing time and connection failure rate. However, this assessment can be categorized as a performance assessment and it lacks the security assessment and analysis. Huang *et al.* [30] introduced a Diversity MTD technique on the virtual servers in order to enhance the resiliency of the network and services. Variations are made to the OS, Visualization components, Web Servers, and application software. Finally, they evaluate their method in terms of the probability of attack success. Azab *et al.* [31] developed a Diversity MTD technique in which it changes the running program's variants erratically. The proposed method is based on dividing a large program into a smaller portion (cells or tasks) that can be performed with several variants (with the same functionality). They also introduced a recovery mechanism to enhance their method's resiliency. Choosing different variant in run-time makes the process of attacker's penetration and scanning difficult. Based on this method, even a successful attack only affects one variant that can be instantly replaced by another variant through a recovery mechanism. However, they considered recovery downtime as their criteria for assessment method and it again lacks security analysis. Moreover, dividing a running application to smaller portions each of which can utilize different changing variant is a complex task and not applicable in any context.

The application of MTD techniques to mitigate DDoS attacks on the cloud using Software-defined networking (SDN) has been studied in many researches [32, 5]. Moreover, Bawany *et al.* [33] conducted a substantial survey to study and classify the proposed SDN-based DDoS attack detection and mitigation techniques. They also proposed an SDN-based proactive DDoS Framework (ProDefence) for detection and mitigation of DDoS attacks in a large-scale network.

Vikram *et al.* [6] proposed a Shuffle MTD technique on the application layer for securing the web by randomizing the HTML elements. Most bots attacking the web uses static HTML elements in the HTTP content/form page. Therefore, randomizing those HTML elements and parameters could be an appropriate technique to mitigate them. A machine learning technique is used to enhance the effectiveness of their defensive strategy. They evaluated their method by measuring the page loading time overhead, and their result reports a low overhead versus the success of thwarting attacks. Machine Learning-based MTD (ML-based MTD) techniques have been proposed in [6, 34]. ML-based MTD enables a defensive system to capture evolving attack patterns with high scalability and applicability. The ML-based techniques can be affected by the lack of a large amount of data for training reasons to provide an acceptable prediction accuracy level. Data-driven incident prediction methods play a crucial role in addressing such ML-based techniques. In [35], the authors surveyed the emerging research for cybersecurity incident prediction focusing on data-driven methodologies. They categorized the related research into six data types category such as the organization's report, dataset, network dataset, synthetic dataset, web page data, social media data, and mixed-type dataset. Anomaly detection techniques and Intrusion Detection System (IDS) can be incorporated with MTD techniques [36]. Network traffic classification is useful methods which can be used for various purposes such as anomaly detection on the networks. In [37] the authors proposed a framework for network traffic classification which can be adapted using very few training samples but high performance. However, this technique can be implemented and used by trigger-based MTD techniques.

Considering MTD techniques in a virtualized environment, Zhang *et al.* [26] proposed an end-to-end defense strategy to secure VMs in a cloud data center at a hypervisor and kernel level. They investigated on VM migrations once an attack is detected, in term of a new location in which a VM can move to. Zhang *et al.* [22] proposed an incentive-compatible MTD method to cope with the problem resulting from co-residency and multi-tenancy (e.g., side-channel attacks) in the virtualized environment [38]. Danev *et al.* [39] proposed a shuffling MTD technique for securing the cloud infrastructure. They focused on VM Migration in the cloud in a secure way. Their approach is to utilize an extra physical Trusted Platform Module, and trusted parties for the migration process. They also used public key infrastructure to secure the protocol. Then devoted a comprehensive evaluation to different criteria like assessing the main Security Services (CIA triad) and performance analysis of the migration scenario in terms of time and RAM size usage against cryptography protocols. Penner and Guirguis [40] developed a set of MTD technologies to change the location of VMs in the cloud in order to defend toward Multi-Armed Bandit attacks caused by weak VM isolation in the cloud. They actually deploy MTD technique based on attacker point of view. They argued that their method avoids Multi-Armed Bandit (MAB) attack designed to find critical information (e.g. dataset and credit card information). They evaluated their method in terms of performance with respect to the time to switch VMs.

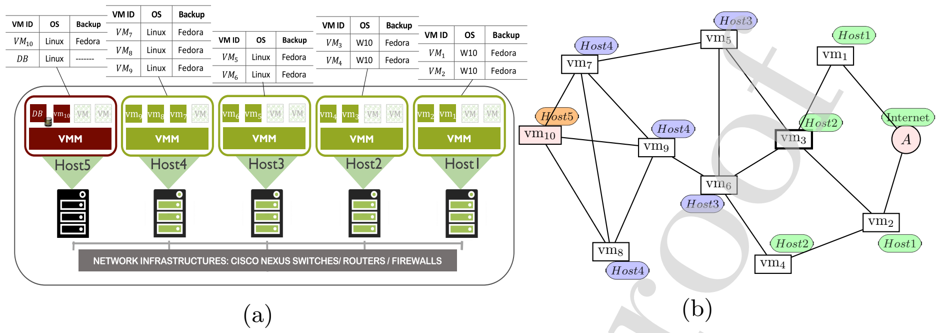
Most of the previous works only focused on the novelty on the proposed strategies and layers of implementation, like defensive method after detecting an attack, low-level shuffling techniques, finding a suitable time-period for applying the IP mutation frequently, dealing with worms and web bots through MTD [6], and so on. However, there are very few works effectively analyzing the MTD techniques for the large networked system needing precise and scalable security analysis, like cloud-based environments. Peng *et al.* [41] investigated the effectiveness of MTD techniques for securing cloud-based services with a heterogeneous or dynamic attack surface. However, they did not utilize a rational or formal security model and analysis tool to evaluate the effectiveness of the deployed strategy. Hong *et al.* [4] analyzed the security changes when MTD techniques are deployed, by introducing a formal method to model Shuffle, Redundancy, and Diversity separately. However, other combinations of MTD techniques should be also considered like deploying a combination of Shuffle and Diversity, and analyzing more security metrics aiming to cover more security requirements of the cloud.

## 3. Preliminaries

In this section, we explain the required notations, concepts and definitions used throughout this paper, such as system setting, configuration and constraints, security metrics, and other related assumptions through a running example in a cloud system.

### *3.1. System and Threat Model*

As a running example for the rest of the paper, we assume a private cloud consisting of five main hosts (servers) each of which can hold up to four active Virtual Machines (VMs). We assume that only two VMs on the first host are allowed to connect to the Internet and only the last host (Host5) is connected to the critical Database (DB), as shown in Figure 1. Each VM includes a default



| VM ID | OS | Backup |
|---|---|---|
| $VM_{10}$ | Linux | Fedora |
| $DB$ | Linux | ------- |

| VM ID | OS | Backup |
|---|---|---|
| $VM_7$ | Linux | Fedora |
| $VM_8$ | Linux | Fedora |
| $VM_9$ | Linux | Fedora |

| VM ID | OS | Backup |
|---|---|---|
| $VM_5$ | Linux | Fedora |
| $VM_6$ | Linux | Fedora |

| VM ID | OS | Backup |
|---|---|---|
| $VM_3$ | W10 | Fedora |
| $VM_4$ | W10 | Fedora |

| VM ID | OS | Backup |
|---|---|---|
| $VM_1$ | W10 | Fedora |
| $VM_2$ | W10 | Fedora |

Figure 1: A Cloud system example: (a) The infrastructure layer of a private cloud. (b) The Cloud example Model showing the connection of the VMs.

operating system (OS) and a backup one tabulated on the top of each server in Figure 1. VMs hosted in the Host1 and Host2 are installed with Windows 10, and VMs in the other hosts are installed with Enterprise Linux OS. We assume that an attacker is outside the private cloud and can exploit the vulnerabilities of those operating systems to gain access. In addition, we also assume that the hypervisors provide enough isolation for VMs hosted on each server. The goal of the attacker is to compromise the Database (DB) in the Host5. The system's configurations, connectivities and constraints are assumed as follows[1]):

- All VMs are active and never suspended
- A VM either can migrate to another host or be launched with its backup OS; the downtime for those processes are negligible
- Two $vm_1$ and $vm_2$ are connected to the Internet (entry points of the cloud)
- $vm_{10}$ on $Host5$ cannot be migrated due to system constraints
- Only $vm_{10}$ can access the Database (DB) which makes the $vm_{10}$ a target to the attackers

Table 1 shows the vulnerabilities for different OS: Win10 (W), Linux (L), and Fedora (F). Here, we only modeled the vulnerabilities that can bypass firewalls and authentications. There are three of those vulnerabilities for both Windows OS and Linux OS, together with a single vulnerability for Fedora OS. Further

[1]These assumptions are used for simplicity in model description and could be released.

Table 1: OS Vulnerabilities ($V$) including Base-Score ($BS$), Impact ($I$), Exploitability ($E$), and Attack Cost ($AC$) 2

| OS ($\theta$) | $V$ | CVE-ID | $BS$ | $I$ | $E$ | $AC$ |
|---|---|---|---|---|---|---|
| **W**in10 | $\nu_{1,\mathrm{W}}$ | CVE-2018-8490 | 8.4 | 6 | 0.17 | 1.6 |
| | $\nu_{2,\mathrm{W}}$ | CVE-2018-8484 | 7.8 | 5.9 | 0.18 | 2.2 |
| | $\nu_{3,\mathrm{W}}$ | CVE-2016-3209 | 8.8 | 5.9 | 0.28 | 1.2 |
| **L**inux | $\nu_{1,\mathrm{L}}$ | CVE-2018-14678 | 7.8 | 5.9 | 0.18 | 2.2 |
| | $\nu_{2,\mathrm{L}}$ | CVE-2018-14633 | 7 | 4.7 | 0.22 | 3 |
| | $\nu_{3,\mathrm{L}}$ | CVE-2017-15126 | 8.1 | 5.9 | 0.22 | 1.9 |
| **F**edora | $\nu_{1,\mathrm{F}}$ | CVE-2014-1859 | 5.5 | 3.6 | 0.18 | 4.5 |

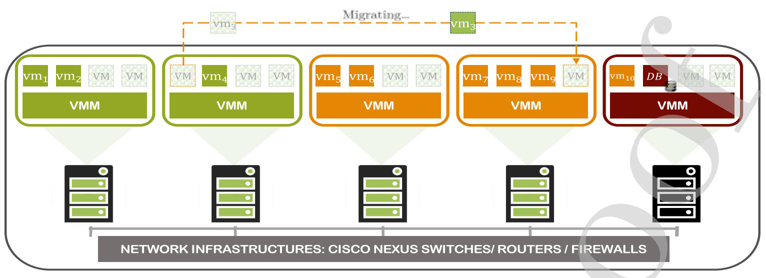
information regarding these vulnerabilities and measures can be found in the National Vulnerability Database (NVD) [42].

### *3.2. Defensive MTD Model*

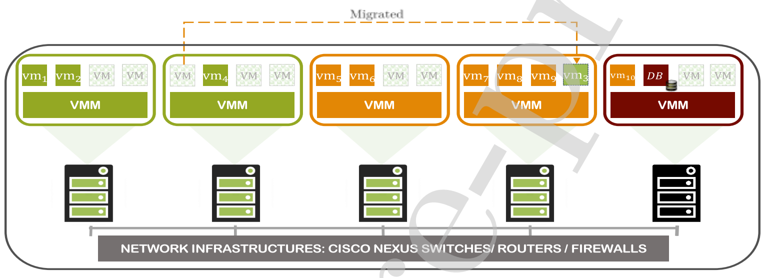
Generation of a defensive model is important for providing appropriate security strategies. The defensive model can analyze the current system and make the decision for deploying the best MTD techniques based on the system requirements, constraints, and limitations. We define a defensive model which can follow the following steps: (1) Model creation (in here, the model we create is HARM for computing the security metrics), (2) Evaluation of current cloud security posture, (3) MTD analyzer to find the best strategies, (4) MTD Deployment on the cloud. We assume that the following operations are permitted by the cloud provider. We then use these operations for deploying MTD techniques.

**Virtual Machine Live Migration:** Virtual Machine Live Migration (VM-LM) can be enabled by the cloud provider, and a VM can migrate from one physical host to another one if there is enough space in the new host. Figure 2 shows an example of VM-LM technique on the cloud example. We use VM-LM as the main approach of deploying Shuffle technique in the cloud. However, other approaches can be used as Shuffle techniques like Virtual IP mutation, Port Hopping , *etc.* [43] which are out of the scope of this paper.

**OS Diversification:** Changing the operating system is the process of using a backup OS instead of the default OS on each VM in the cloud. OS


Migrating...
vm3
VM
vm1
vm2
vm4
vm5
vm6
vm7
vm8
vm9
vm10
DB
VMM
NETWORK INFRASTRUCTURES: CISCO NEXUS SWITCHES/ ROUTERS / FIREWALLS

(a) Migrating


Migrated
VM
vm1
vm2
vm4
vm5
vm6
vm7
vm8
vm9
vm3
vm10
DB
VMM
NETWORK INFRASTRUCTURES: CISCO NEXUS SWITCHES/ ROUTERS / FIREWALLS

(b) Migrated

Figure 2: VM-Live Migration on the Cloud. (a) Migrating Phase. (b) Migrated.

Diversification can be used as a Diversity technique [18]. We assume that the probability of failure in launching a new OS is negligible. Changing the OS may introduce a new set of vulnerabilities to the attacker. Other Diversity methods can be utilized such as changing applications and services running on the VMs, changing programming languages, etc. However, in this paper we only consider OS Diversification to deploy Diversity.

We deploy VM-LM and OS Diversification through simulation. In this study, the focus is not on implementation on the real cloud but the theoretical appraisal and formalism, and only considered theoretical analysis and evaluation through simulation to assess the effectiveness of combining MTD techniques. However, the feasibility, adaptability, and, usability of those MTD techniques on a private cloud, named UniteCloud [44], are practically considered in [45].

## *3.3. Graphical Security Models (GSM)*

HARM is proposed by Hong *et al.* which is a scalable GSM for analyzing the security of the systems through two layers. HARM has been used in many studies needing scalable security analysis like enterprise security analysis [46, 47], cloud computing, IoT, and MTD [12, 4]. In this paper, we use HARM for security analysis and evaluate the MTD techniques. HARM separates the networks' connectivities and vulnerabilities into two layers so that reachability of the network's components is captured in the upper layer and the vulnerabilities existing on each component can be captured in the lower layer. We utilize HARM to capture the connectivities of VMs on the cloud in the upper layer and OS vulnerabilities excising on each VM in the lower layer of HARM. Constructing the HARM, we can perform the security analysis and compute the security metrics.

**Definition 1.** We can show HARM [14] as a 3-tuple $H = (U, L, C)$ where $U$ refers to the upper layer which is an Attack Graph (AG), and $L$ represents the lower layer in which an Attack Tree (AT) is constructed. We define $C = U \to L$ as a one-by-one mapping of the upper layer to the lower layer.

**Definition 2.** The upper layer of HARM is defined as a graph $U = (\mathrm{VM}, \mathrm{E})$, where $\mathrm{VM} = \{vm_1, vm_2, \ldots, vm_n\}$ is a set of VMs in the cloud, with $|VM| = n$, and $E \in VM \times VM$ is a set of connectivities between VMs.

**Definition 3.** The lower layer $L$ is a set of ATs corresponding to each VM $vm_i$ in the upper layer and can be defined as $L = \{\ell_1, \ell_2, \ldots, \ell_n\}$, where $\ell_i = (V_{i,\theta}, G, root)$, and $\ell_i$ is an AT corresponding to the $vm_i$. Then, $V_{i,\theta} = \{\nu_{1,\theta}, \nu_{2,\theta}, \ldots, \nu_{m,\theta}\}$ is a set of vulnerabilities existing on each corresponding OS $\theta \in OS$ on each VM $vm_i$, where $OS = \{W, L, F\}$. We denote the number of vulnerabilities in each VM (specifically, on each OS) as $|V_{i,\theta}| = m$, and $G$ is a set of logical gates $G = \{AND\text{-gate}, OR\text{-gate}\}$ constructing the inner nodes of the AT, and $root$ is the corresponding node in $U$.

Figure 3a demonstrates the generated HARM for the cloud system example presented in Figure 1. We can calculate the security metrics including R, AC, RoA using the generated HARM. We then can compute and evaluate the overall security of the cloud by considering both sides: the cloud provider and the

Table 2: Notations, explanations and supporting examples or references

| Notations | | Description | Formula, Reference |
|---|---|---|---|
| | | General | |
| $VM$ | | The set of all Virtual Machines (VMs) in the cloud, $\|VM\| = n$ | $VM = \{vm_1, vm_2, \ldots, vm_n\}$ |
| $V_{i,\theta}$ | | The set of all vulnerabilities existing on $i^{th}$ VM, i.e. $vm_i$, $\|V_{i,\theta}\| = m$ | $V_{i,\theta} = \{\nu_{1,\theta}, \nu_{2,\theta}, \ldots, \nu_{m,\theta}\}$ |
| $\nu_{j,\theta}$ | | The $j^{th}$ vulnerability in a single VM, $\nu_{j,\theta} \in V_{i,\theta}$ | Table 1 |
| $I_{\nu_{j,\theta}}$ | | The impact of exploiting the $j^{th}$ vulnerability in a single VM, $\nu_{j,\theta} \in V_{i,\theta}$. | Table 1 |
| $E_{\nu_{j,\theta}}$ | | The exploitability of the $j^{th}$ vulnerability in a single VM, $\nu_{j,\theta} \in V_{i,\theta}$. | Table 1 |
| $ap$ | | An attack path in the cloud defined as a series of adjacent VMs in the cloud from the entry point to the target. | $ap = (vm_1, vm_2, \ldots, vm_n)$ |
| $AP$ | | All possible attack paths in the cloud from an entry point to a target, $\|AP\| = p$ | $AP = \{ap_1, ap_2, \ldots, ap_p\}$ |
| $C_b(vm_i)$ | | A function returning the betweenness value of a VM in the cloud. | Equation 1 |
| | | Security Metrics | |
| Cloud Risk ($R_c$) | $R_{vm_i}$ | The risk value associated with a VM in the cloud | Equation 4 |
| | $R_{ap}$ | The count of all VMs' risk values of an attack path $ap$ in the cloud | Example 2 |
| | $R_c$ | The total risk of the cloud based on an entry point and the target | Equation 4 |
| Attack Cost ($AC_c$) | $AC_{vm_i}$ | The cost of exploiting a VM in the cloud for an attacker by considering attacker's knowledge and excising vulnerabilities in the VM. | Table 1 |
| | $AC_{ap}$ | The count of all VMs' AC values of an attack path $ap$ in the cloud | Example 5 |
| | $AC_c$ | The total attack cost imposed on an attacker to successfully compromise the target in the cloud | Equation 5 |
| Return on Attack ($RoA_c$) | $RoA_{vm_i}$ | The total gain an attacker may achieve by compromising a VM in the cloud against the attacker's efforts | Equation 6 |
| | $RoA_{ap}$ | The count of all VMs' RoA values of an attack path $ap$ in the cloud | Equation 7 |
| | $RoA_c$ | The total benefits an attacker gain by successfully compromising the target in the cloud by considering the attacker's effort | Equation 7 |

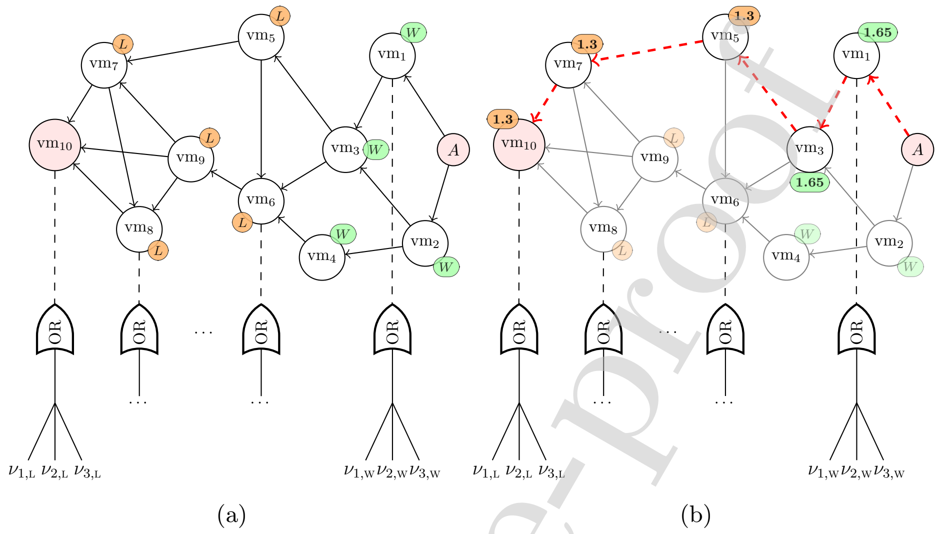


Figure 3: (a) Two-layer HARM of the Cloud example (b) An attack path from the attacker to target in HARM (the risk values of exploiting each VM are depicted beside each VM)

attackers. In the upper layer of the HARM, an AG is used, and in the lower layer an AT is used, such as in [4]. Figure 3a shows the constructed HARM for the example cloud system.

Hence, the generated HARM for the cloud system illustrated in Figure 1 is presented as follows.

**Example 1.** The HARM for the Cloud system example (cs) is shown as $H_{cs} = (U_{cs}, L_{cs}, C_{cs})$. Then, $U_{cs} = (VM_{cs}, E_{cs})$, where $VM_{cs} = (A, vm_1, vm_2, \dots, vm_n)$. The VMs are connected as $E_{cs} = \{(A, vm_1), (A, vm_2), (vm_1, vm_3), (vm_2, vm_3), (vm_2, vm_4), (vm_3, vm_5), (vm_3, vm_6), (vm_4, vm_6), (vm_5, vm_6), (vm_5, vm_7), (vm_6, vm_9), (vm_7, vm_8), (vm_7, vm_{10}), (vm_8, vm_{10}), (vm_9, vm_7), (vm_9, vm_8), (vm_9, vm_{10})\}$. Then, $L_{cs} = \{\ell_1, \ell_2, \dots, \ell_{10}\}$. For example, $\ell_1 = \{V_{1,W}, G_1, root_1\}$ shows the AT for $vm_1$, where $V_{1,W}$ is a set of OS vulnerabilities (Windows) existing on $vm_1$, and $G_1 = OR - gate$, and $root_1 = vm_1$.

### *3.3.1. Importance Measures (IM)*

As stated earlier, security analysis through GSM suffers from scalability problems, especially, when we use Exhaustive Search (ES) to find the optimal solution. To address this shortfall, we utilize Betweenness Centrality Measures ($C_b$) [48] as the IM to discover more critical VMs in the cloud. Using IMs we can find the most important nodes in the network (VMs in here) in the upper layer of HARM. Then we deploy our MTD Technique on a set of crucial VMs without using ES. In our previous study [11], we showed that the best MTD scenario could be found by deploying MTD on the VMs having higher values of Betweenness in the cloud. In fact, Betweenness is a graph metric which shows the nodes lying on the paths between other network's node and $C_b$ of a VM can be defined and computed as Definition (4).

**Definition 4.** We formulate Network Centrality Measures (NCM) based on the upper layer of HARM defined in Definition 2. let $\delta_{st}$ be a function calculating the total number of shortest path between each pair of VMs $(s,t) \in VM$, and $\delta_{st}(vm_i)$ denotes the number of those paths passing through the specific VM $(vm_i)$.

$$C_b(vm_i) = \sum_{s,t \in VM \setminus \{vm_i\}} \frac{\delta_{st}(vm_i)}{\delta_{st}}, \tag{1}$$

Constructing the HARM, we can find the most important VM in the cloud based on the HARM definition denoted as $\kappa = \Gamma(H)$, where $\Gamma(H)$ is a function returning the argument of a VM in the HARM $(H)$ having the highest Betweenness values based on Equation 2.

$$\Gamma(H) = \arg \max_{vm_i \in VM} C_b(vm_i) \tag{2}$$

Then, the value of $\kappa$ determines the argument of the VM that needs to be selected for deploying MTD technique on. For instance, $\kappa = 3$ shows that MTD technique should be applied on $vm_3$.

### *3.3.2. Cloud Risk*

Constructing the HARM, we can compute the security metrics. The overall risk value associated with the cloud is called Cloud Risk ($R_c$). We can construct the HARM through obtaining information related to the VM connectivities on the cloud and considering the vulnerabilities of each VM through a vulnerability database (i.e. NVD) as listed in Table 1. In order to measure the $R_c$, we use both layers of HARM in which VMs connectivities are presented on the upper layer and vulnerabilities are captured on the lower layer. Let $E_{\nu_{j,\theta}}$ and $I_{\nu_{j,\theta}}$ be the exploitability and impact values of $j^{th}$ vulnerability existing on the OS $\theta$ on the $vm_i$ such that $\nu_{j,\theta} \in V_{i,\theta}$. Then, the risk of a VM can be computed as the product of those values returning the maximum value, as shown in Equation 3.

$$R_{vm_i} = \max_{\nu_{j,\theta} \in V_{i,\theta}} \left( E_{\nu_{j,\theta}} \times I_{\nu_{j,\theta}} \right). \tag{3}$$

Thus, the value of $R_{ap}$ can be computed as counting all risk values on any single VM in an attack path ($ap$) from the attacker to the target. Then, the sum of all risk values associated with all possible attack paths on the system provides the $R_c$ value, see Equation 4.

$$R_c = \sum_{ap \in AP} \left( \sum_{vm_i \in ap} R_{vm_i} \right) \tag{4}$$

**Example 2.** Figure 3b shows an attack path from the attacker to target and can be shown as $ap_1 = \{(A, vm_1), (vm_1, vm_3), (vm_3, vm_5), (vm_5, vm_7), (vm_7, vm_{10})\}$, then $R_{ap_1} = R_{vm_1} + R_{vm_3} + R_{vm_5} + R_{vm_7} + R_{vm_{10}} = 5.9 * 0.28 + 5.9 * 0.28 + 5.9 * 0.22 + 5.9 * 0.22 + 5.9 * 0.22 = \mathbf{7.2}$. Finally, the total risk of the system is as sum of all R$_{ap}$ values which is $R_c = \mathbf{211.692}$ for the cloud example.

### *3.3.3. Attack Cost*

The cost of exploiting the vulnerabilities on a VM by an attacker is defined as Attack Cost ($AC$). We expand this metric to compute the overall $AC$ of the cloud. We use the upper layer of HARM to compute $AC$. Table 1 lists the cost of exploiting a VM through vulnerabilities ($AC_{vm}$). Equation 5 shows the calculation formula for the overall $AC$ value of the cloud.

$$AC_c = \sum_{ap \in AP} \left( \sum_{vm_i \in ap} AC_{vm_i} \right) \tag{5}$$

where $ap$ is a single attack path in the system and $AP_s$ is the list of all possible attack paths in the network.

### *3.3.4. Return on Attack*

Return on Attack (RoA) is another security metrics from the attacker's perspective [49]. RoA quantifies the attack cost versus benefits of attack. The higher value of RoA indicates a higher probability that attacker exploit those vulnerabilities (higher tendency to attack). The ratio of risk value for a VM and attack cost determines the RoA value for a specific VM which are shown in Equation 6. Then, the overall RoA of a system can be computed through Equation 7.

$$RoA_{vm_i} = \frac{\max_{\nu_{j,\theta} \in V_{i,\theta}} \left( E_{\nu_{j,\theta}} \times I_{\nu_{j,\theta}} \right)}{AC_{vm_i}} \tag{6}$$

$$RoA_c = \sum_{ap \in AP} \left( \sum_{vm_i \in ap} RoA_{vm_i} \right) \tag{7}$$

### *3.3.5. Path-based Metrics*

We consider path-based metrics as presented in [50] to quantify the network security level resulting from deploying MTD techniques on each VM. We use four path-based metrics in this paper, Shortest Attack Path (SAP), Mean of Attack Path Lengths (MAPL), Standard Deviation of Path Lengths (SDPL), Mode of Path Lengths (MoPL), but other metrics such as Attack Resistance Metric, Network Compromise Percentage [50] can also be computed. Path-based metrics only considers the reachability of the network and the changes on the vulnerabilities in the VMs may not affect the path-based metrics. Thus, Path-based metrics may be changed if only the upper layer of HARM is changed, and the changes in the lower layer of HARM resulting from adding or removing the vulnerabilities have no effects on the path-based metrics. The SAP metric for the cloud example, as shown in Figure 3a can be computed as Equation 8.

$$SAP_c = \min_{ap \in AP} |ap| = 5 \tag{8}$$

The $SAP_c$ value indicates the minimum number of VMs the attacker needs to exploit before compromising the target VM. The cloud provider may select the appropriate countermeasure to increase $SAP_c$ to make the attack more difficult for the attacker. For example, the cloud provider may choose a replacement strategy for Shuffle so that a VM can be injected to the shortest path to make the attack path more difficult to exploit.

The existence of more number of attack paths ($p$) in the network causes less security as the attacker can leverage different alternative paths to launch the attacks. Thus, the increment of $p$ and higher values for MAPL indicates less security in the network. Equation 9 shows the calculation for MAPL for the cloud example. Moreover, the SDPL metric for the cloud example can be computed as Equation 10.

$$MAPL_c = \frac{\sum_{ap \in AP} |ap|}{p} = 6.25 \tag{9}$$

$$SDPL_c = \sqrt{\frac{\sum_{ap \in AP} (|ap| - \text{MAPL}_c)^2}{p}} = 0.89 \tag{10}$$

### *3.3.6. Attack Success Probability*

We can generate the ATs in the lower layer of HARM based on the given vulnerabilities and compute Attack Success Probability (ASP) for each VM $vm_i$ in the cloud denoted as $ASP_{vm_i}$. However, generating ATs from vulnerabilities needs a clear understanding of the vulnerabilities and the way in which they can be exploited. For instance, an attacker can exploit only a single vulnerability to penetrate into a VM, or the attacker may need to exploit a set of vulnerabilities to penetrate into a VM. In the former, a logical OR-gate can be used and for the latter, a combination of logical AND/OR-gates can be used as presented

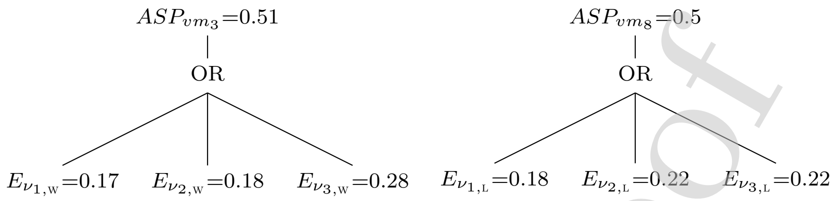


Figure 4: Generating Lower Layers AT to calculate ASP for $vm_3$ and $vm_8$.

in [51]. For simplicity, we assume that the attacker can compromise a VM by exploiting any vulnerability existing in a VM. In this case, only OR-gate is used to generate AT. The $ASP_{vm_i}$ can be obtained based on the exploitability values $(E)$ of the vulnerabilities on the VM. Let $E_{v_{j,\theta}}$ be the exploitability value of $j^{th}$ vulnerability existing on the OS $\theta$ on the $vm_i$ such that $v_{j,\theta} \in V_{i,\theta}$. Then, we can compute the $ASP_{vm_i}$ as Equation 11.

$$ASP_{vm_i} = 1 - \prod_{v_{j,\theta} \in V_{i,\theta}} \left(1 - E_{v_{j,\theta}}\right) \tag{11}$$

In this paper, we only consider the OS vulnerabilities. For example, we assume that there are three vulnerabilities for Win10 as shown in Table 1 and $vm_3$ uses Win10. Thus, based on the Example 1, the lower layer of HARM for $vm_3$ can be shown as $\ell_3 = \{V_{3,\mathrm{W}}, G_3, root_3\}$ shows the AT for $vm_3$, where $V_{3,\mathrm{W}}$ is a set of Win10 vulnerabilities existing on $vm_3$, and $G_3 = OR - gate$, and $root_3 = vm_3$. Then, the $ASP_{vm_3}$ can be computed as $1 - (1 - 0.17) \times (1 - 0.18) \times (1 - 0.28) = 0.51$. The generated ATs for both $vm_3$ and $vm_8$ with the corresponding ASP values are shown in Figure 4. Finally, in order to compute the total ASP for the cloud $ASP_c$, we used SHARPE (Symbolic Hierarchical Automated Reliability and Performance Evaluator) [52], which uses a reliability graph to quantify the overall ASP. The upper layer of HARM including the ASP for each VM can be fed into the SHARPE as a reliability graph.

Table 3: Notations w.r.t deploying MTD techniques and metrics

| **Symbol** | **Descriptions** |
|---|---|
| $S \otimes D$ | Combination of Shuffle ($S$) and Diversity ($D$) strategies |
| $S{+}D$ | Strategy: deploying $S$ first, then $D$ |
| $S \Delta D$ | Strategy: deploying $S$ first, computing IM, then $D$ |
| $R_c^S$ | Total cloud's risk after deploying $S$ only |
| $R_c^D$ | Total cloud's risk after deploying $D$ only |
| $R_c^{S+D}$ | Total cloud's risk after deploying $S{+}D$ |
| $R_c^{S\Delta D}$ | Total cloud's risk after deploying $S \Delta D$ |
| $AC_c^S$ | Total attack cost value after deploying $S$ only |
| $AC_c^D$ | Total attack cost value after deploying $D$ only |
| $AC_c^{S+D}$ | Total attack cost value after deploying $S{+}D$ |
| $AC_c^{S\Delta D}$ | Total attack cost value after deploying $S \Delta D$ |
| $RoA_c^S$ | Total return on attack value after deploying $S$ only |
| $RoA_c^D$ | Total return on attack value after deploying $D$ only |
| $RoA_c^{S+D}$ | Total return on attack value after deploying $S{+}D$ |
| $RoA_c^{S\Delta D}$ | Total return on attack value after deploying $S \Delta D$ |
| $ASP_c^S$ | Attack Success Probability after deploying $S$ only |
| $ASP_c^D$ | Attack Success Probability after deploying $D$ only |
| $ASP_c^{S+D}$ | Attack Success Probability after deploying $S{+}D$ |
| $ASP_c^{S\Delta D}$ | Attack Success Probability after deploying $S \Delta D$ |

## 4. MTD Techniques Deployment

Deploying MTD techniques on the cloud depends on the constraints defined by the cloud providers. For instance, some operations may be restricted by the cloud providers such as VM-LM from a host to a specific host (which is protected). In this paper, we assume that VM-LM and OS Diversification are allowed by the cloud provider. Then, we utilize VM-LM and OS Diversification techniques to develop Shuffle and Diversity techniques respectively. Later on in Section 5, we discuss the MTD combination deployment strategies. In order to evaluate the effects of MTD techniques on the cloud, we analyze the cloud security level before deploying MTD techniques by computing the security metrics: $R_c$, $AC_c$, and $RoA_c$ as described in Table 2, and then analyze the effectiveness of each selected MTD technique by reconsidering those security metrics again, the notations and symbols are presented in Table 3. In this section, we define and formalize the MTD techniques we used in this paper which are Shuffle, Diversity, and the combinations of both. Then, we consider the effects of deploying each MTD techniques on the security metrics.

### 4.1. Shuffle Technique Definition and Formalization

In general, Shuffle techniques work based on the rearrangement of the system's components, settings, and configuration such as mutating or shuffling the IP address, re-arranging the network's topology, moving a VM from one host to another one, and *etc.* [5, 39, 43]. Shuffle strategies can be deployed in different layers like Application Layer, Virtualization, Host, Virtual Machine Monitoring (VMM), Hardware [6, 22, 53]. In this paper, we deploy MTD techniques on the virtualization layer. We utilize VM-LM method to deploy Shuffle in our simulation, as depicted in Figure 2. Migration of each VM from a host to another machine may affect the overall security of the system. For this reason, before deploying Shuffle technique, we investigate the effects of deploying Shuffle technique on the cloud by analyzing the security metrics which are $R_c$, $AC_c$, and $RoA_c$ through HARM.

Deploying Shuffle techniques falls into two scenarios: (1) Selecting the VM for deploying Shuffle which is usually named as 'what to move' in the literature [54]. (2) The place in which the VM should be moved into which is called as 'where to move' [54]. In this paper, we utilized the both Exhaustive Search (ES) and Important Measures (IMs) for the former scenario and a replacement algorithm for the latter scenario which chooses the shortest path in the upper layer of the HARM and migrates the selected VM to that place. This replacement algorithm aims to increase the length of the shortest attack path which attacker can pass through from the entry point (the Internet) to the target (i.e. a DB).

As the Shuffle technique only changes the physical locations of VMs (i.e. from a physical server in the cloud to another server), it only affects the reachabilities of VMs in the upper layer of HARM. Shuffle technique can be formulated based on the HARM definition as follows.

**Definition 5.** Let $S(H, \kappa)$ be a Shuffle function on the HARM where $1 \leq \kappa \leq n$, and $\kappa$ denotes a specific VM $vm_\kappa \in VM$ chosen for the shuffling. Then the result of Shuffle function is as $S(H, \kappa) = H^{\mathrm{s}}$. We define $H^{\mathrm{s}} = (U^{\mathrm{s},\kappa}, L, C)$ where $U^{\mathrm{s},\kappa}$

Table 4: The percentage of changes in the security metrics after deploying Shuffle on each VM in the cloud example

| ID | Values | | | % of Changes | | |
|---|---|---|---|---|---|---|
| | $R_c^S$ | $AC_c^S$ | $RoA_c^S$ | $R_c^S$ | $AC_c^S$ | $RoA_c^S$ |
| vm$_1$ | 164.49 | 201.8 | 102.81 | -22.30% | -19.73% | -24.28% |
| vm$_2$ | 130.51 | 161.8 | 80.86 | -38.35% | -35.64% | -40.44% |
| vm$_3$ | **82.25** | 100.9 | **51.40** | **-61.15%** | -59.86% | **-62.14%** |
| vm$_4$ | 174.05 | 209.7 | 110.37 | -17.78% | -16.59% | -18.70% |
| vm$_5$ | 116.94 | 139.5 | 74.73 | -44.76% | -44.51% | -44.95% |
| vm$_6$ | 88.26 | 102.4 | 57.61 | -58.31% | -59.27% | -57.56% |
| vm$_7$ | 122.48 | 146.4 | 78.16 | -42.14% | -41.77% | -42.43% |
| vm$_8$ | 205.67 | 236.5 | 135.13 | -2.84% | -5.93% | -0.46% |
| vm$_9$ | 93.1 | 110.7 | 59.65 | -56.02% | -55.97% | -56.06% |

is the transformed AG resulted from Shuffle on $vm_k$ in the upper layer of the HARM and can be represented as $U^{\mathrm{s},\kappa} = (VM, E')$, where $E' \subseteq VM \times VM$.

### *4.2. Shuffle Technique Evaluation*

According to the Shuffle definition 5, the result of deploying Shuffle on the cloud example (Figure3a) is as follows.

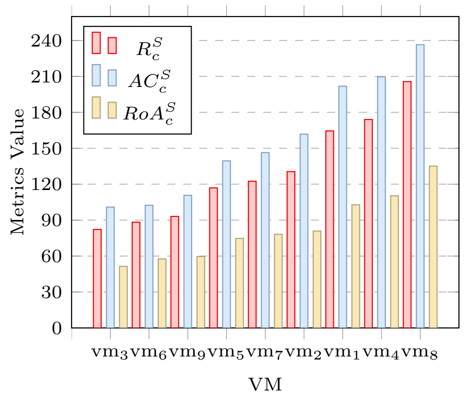
**Example 3.** The result of Shuffle function on the generated HARM for the cloud example ($H_{cs}$) in which the most crucial VM is selected to be shuffled ($vm_\kappa$) can be represented as $S(H_{cs}, \kappa) = H^s_{cs}$. Based on Equation 2, $\kappa = 3$ which means $vm_3$ is selected as a VM with the highest IM values on the cloud. Then, $H^s_{cs} = (U^{\mathrm{s},3}_{cs}, L, C)$ which $U^{\mathrm{s},3}_{cs} = (VM_{cs}, E'_{cs})$ is the upper layer of HARM for cloud system, where $VM_{cs} = \{A, vm_1, vm_2, vm_3, vm_4, vm_5, vm_6, vm_7, vm_8, vm_9, vm_{10}\}$, and $E'_{cs}$ denotes the connectivity of VMs after deploying Shuffle technique on $vm_3$ and is represented as $E'_{cs} = \{(A, vm_1)$,$(A, vm_2)$, $(vm_2, vm_3)$, $(vm_3, vm_4)$, $(vm_4, vm_6)$, $(vm_6, vm_9)$, $(vm_5, vm_6)$, $(vm_5, vm_7)$, $(vm_7, vm_{10})$, $(vm_9, vm_8)$,$\ldots\}$. Note that Shuffle technique only changes the upper layer of HARM and preserves $L$ and $C$.

Based on the definition of the Shuffle technique 5, we deploy Shuffle on a VM and then recalculate the security metrics to observe the effectiveness of the deployed technique in the cloud. For Example, deploying $S(H_{cs}, vm_3)$ may add/remove edges in the upper layer of HARM. Let consider a new attack path resulting from deploying Shuffle $ap_1 = \{(A, vm_2), (vm_2, vm_3), (vm_3, vm_4), (vm_4, vm_6), (vm_6, vm_9), (vm_9, vm_8), (vm_8, vm_{10})\}$, then we can compute the $R_{ap_1} = R_{vm_2} + R_{vm_3} + R_{vm_4} + R_{vm_6} + R_{vm_9} + R_{vm_8} + R_{vm_{10}} = 1.65 + 1.65 + 1.65 + 1.3 + 1.3 + 1.3 + 1.3 = \mathbf{10.15}$. Likewise, we calculate the $AC_{ap_1}$, $RoA_{ap_1}$ as follows. $AC_{ap_1} = AC_{vm_2} + AC_{vm_3} + AC_{vm_4} + AC_{vm_6} + AC_{vm_9} + AC_{vm_8} + AC_{vm_{10}} = 1.2 + 1.2 + 1.2 + 1.9 + 1.9 + 1.9 + 1.9 = \mathbf{11.2}$. $RoA_{ap_1} = RoA_{vm_2} + RoA_{vm_3} + RoA_{vm_4} + RoA_{vm_6} + RoA_{vm_9} + RoA_{vm_8} + RoA_{vm_{10}} = 1.38 + 1.38 + 1.38 + 0.68 + 0.68 + 0.68 + 0.68 = \mathbf{6.86}$.

We denote the overall security metrics after deploying Shuffle techniques on the cloud as $R_c^S$, $AC_c^S$, and $RoA_c^S$. Then, after deploying Shuffle on $vm_3$, $R_c^S = \mathbf{82.25}$, $AC_c^S = \mathbf{100.9}$, and $RoA_c^S = \mathbf{51.4}$. We then calculate the results of deploying Shuffle on other VMs and compare them. Figure 5a reveals the results of deploying Shuffle technique on each VM in the cloud example. We illustrate the VMs sorted in descending order based on their IMs values in the x-axis and the security metrics values for $R_c^S$, $AC_c^S$, $RoA_c^S$ in the y-axis. It is obvious that the values obtained for VMs $vm_3$, $vm_6$, and $vm_9$ (which have higher ranks in terms of IMs) are lower than the others, while deploying Shuffle on the VMs $vm_4$ and $vm_8$ (which have lower IMs ranks) yields higher $R_c$ and $RoA_c$. Accordingly, we tabulated the percentage of the changes in the security metrics in Table 4. The results show that the $R_c^S$, and $RoA_c^S$ are improved after deploying Shuffle techniques on the cloud and the best percentage of changes belongs to deploying Shuffle on $vm_3$ with **-61.15%** and **-62.14%** for $R_c^S$ and $RoA_c^S$, respectively. The lower values of $RoA_c$ shows that the attacker has less orientation to launch the attacks again on that specific cloud's configuration, while the higher rate of the $RoA_c$ shows the higher probability of the same attack is launched by the attacker.

### *4.3. Diversity Technique Definition and Formalization*

The main idea behind the Diversity techniques is to replace the systems' components (e.g. a VM, server, programming language, OS, hardware, and


(a) $S$

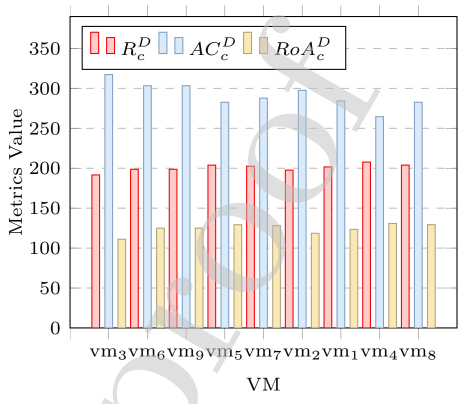

(b) $D$

Figure 5: Comparison of security metrics after deploying two Shuffle MTD and Diversity MTD strategies separately on the VMs (VMs in x-axis are sorted in descending order based on their importance using IMs). (a) The results of deploying Shuffle technique on the cloud example. (b) The results of deploying Diversity technique on the cloud example.

*etc.*) with different variants or implementations, while the system provides the same services for the customers[55, 9]. We change the OS instances (OS Diversification) in the cloud to deploy Diversity technique. Unlike Shuffle technique which may change the reachability of the VMs in the cloud, deploying Diversity has no effects on the reachability of the VMs and may vary the vulnerabilities existing on the VMs (only the lower layer of HARM may be affected and the upper layer remains the same). In this study, we only consider OS vulnerabilities though other vulnerabilities can also be investigated through HARM [56]. Deploying Diversity has two advantages. First, it invalidates the information collected by the attacker to exploit the vulnerabilities of the current system. Secondly, Diversity can introduce new vulnerabilities to the attacker, thus the adversary needs to deal with new vulnerabilities and spend more time and cost to gain information and exploit them. In this paper, we assume that the cloud provider can use a backup OS which imposes a higher cost to the attacker based on the vulnerabilities on the new OS, we consider AC as the difficulties to exploit vulnerabilities using NVD database. Diversity can be formulated based on the HARM definition as follows.

**Definition 6.** We formulate the Diversity technique in which the diversity function is applied on $H$ as $D(H, \kappa) = H^d$, where $\kappa$ denotes a specific VM $vm_\kappa \in VM$ selected for replacing with another OS variant. Then, $H^d = (U, L^{d,k}, C)$ is the result of deploying Diversity technique, where $L^{d,k} = \{\ell_1, \ldots, \ell_{\mathbf{k}}, \ldots, \ell_n\}$ denotes the ATs corresponding to each VM and $\ell_k = (V_{k,\theta}, G, root)$ is the transformed AT of $vm_k$ which is replaced with another variant $\theta \in OS$. Diversity technique affects the lower layer and varies vulnerabilities $V_{k,\theta} = \{\nu_{1,\theta}, \nu_{2,\theta}, \ldots, \nu_{m,\theta}\}$, while $U = (VM, E)$ is preserved.

## *4.4. Diversity Technique Evaluation*

In order to analyze Diversity technique, we first create the HARM for the cloud example 1 and compute the security metrics to observe the current security posture of the cloud. Then, we deploy the Diversity technique on the VMs based on the Diversity definition 6. Finally, we compare and analyze the security metrics after deploying each Diversity scenario. The following example shows the Diversity based on the given Diversity definition 6. We change the current OS for each VM with the backup one. To be specific, we use Fedora as the backup OS which includes different vulnerability values, see Table 1.

**Example 4.** The result of Diversity function on the HARM for the cloud example ($H_{cs}$) in which a VM is selected to be shuffled ($vm_\kappa$) is represented as $S(H_{cs}, \kappa) = H^d_{cs}$. For example, we select $vm_3$ having the highest IM value on the cloud for deploying Diversity. Then, $H^d_{cs} = (U_{cs}, L^{d,3}_{cs}, C)$ which $U_{cs} = (VM_{cs}, E_{cs})$ remains unchanged, and $\ell_3 = (V_{3,W}, G, root)$, where $\ell_3 \in L^{d,3}_{cs}$, and $V_{3,W} = \{\nu_{1,W}, \nu_{2,W}, \nu_{3,W}\}$, has been replaced with the backup OS (Fedora(F)). Then, $\ell_3 = (V_{3,F}, G, root)$ where, $V_{3,F} = \{\nu_{1,F}\}$ as in Table 1

Unlike Shuffle, deploying Diversity will not add/remove or change any attack path in the upper layer of the HARM, but it varies the lower layer of the HARM by introducing different vulnerabilities. Consequently, the security metrics regarding each attack path may change. For example, let consider an attack path existing on the cloud example (showed by red dashed lines in Figure 3b) which is

Table 5: The percentage of changes in the security metrics after deploying Diversity on each VM in the cloud example

| ID | Values | | | % of Changes | | |
|---|---|---|---|---|---|---|
| | $R_c^D$ | $AC_c^D$ | $RoA_c^D$ | $R_c^D$ | $AC_c^D$ | $RoA_c^D$ |
| vm$_1$ | 201.65 | 284.4 | 123.44 | -4.74% | 13.13% | -9.08% |
| vm$_2$ | 197.64 | 297.6 | 118.51 | -6.64% | 18.38% | -12.71% |
| vm$_3$ | **191.61** | **317.4** | **111.11** | **-9.5%** | **26.25%** | **-18.16%** |
| vm$_4$ | 207.68 | 264.6 | 130.83 | -1.9% | 5.25% | -3.63% |
| vm$_5$ | 203.89 | 282.6 | 129.29 | -3.68% | 12.41% | -4.77% |
| vm$_6$ | 198.69 | 303.4 | 124.98 | -6.14% | 20.68% | -7.94% |
| vm$_7$ | 202.59 | 287.8 | 128.21 | -4.3% | 14.48% | -5.56% |
| vm$_8$ | 203.89 | 282.6 | 129.29 | -3.68% | 12.41% | -4.77% |
| vm$_9$ | 198.69 | 303.4 | 124.98 | -6.14% | 20.68% | -7.94% |

defined as $ap_1 = \{(A, vm_1), (vm_1, vm_3), (vm_3, vm_5), (vm_5, vm_7), (vm_7, vm_{10})\}$. Then, we deploy Diversity techniques on $vm_3$, and calculate the security metrics for $ap_1$ as follows. $R_{ap_1} = R_{vm_1} + R_{vm_3} + R_{vm_5} + R_{vm_7} + R_{vm_{10}} \approx \mathbf{6.2}$. Likewise, $AC_{ap_1}$ and $RoA_{ap_1}$ are around **11.4** and **3.56** respectively. Then, the overall values of the security metrics after deploying Diversity on $vm_3$ are as follows: $R_c^D = \mathbf{191.61}$, $AC_c^D = \mathbf{317.4}$, $RoA_c^D = \mathbf{111.11}$. To compare with the other deployment scenarios, we compute the Diversity technique on the other VMs in the cloud example and compare the results among them. Figure 5b shows the values of Diversity on each VM in the cloud example. The VMs are sorted in descending order based on their IMs values on the x-axis and the security metrics on the y-axis. The results show that deploying Diversity highly increases the $AC$ value. However, the highest $AC$ values can be found through deploying Diversity on the VMs $vm_3$, $vm_6$, and $vm_9$ which have higher IMs values. Similarly, better results can be obtained for $R$ and $RoA$ values for $vm_3$. Table 5 lists the percentage of changes in the security metrics after deploying Diversity showing the negative changes for $R$ and $RoA$ values and the positive changes on $AC$ values. Although changing OS is used for deploying Diversity technique, the variation on the security metrics such as $AC$ highly depend on the vulnerability information introduced by the new OS.

## 5. MTD Combinations Definition and Formalization

In this section, we investigate the effects of deploying the combination of Shuffle and Diversity together using the security metrics. Based on the results reported in sections 4.2 and 4.4, deploying Shuffle technique decreases both $R_c^S$ and $RoA_c^C$, but it also reduces the $AC_c^S$ values. However, deploying Diversity increases $AC_c^S$ values, but the percentage of changes in $R_c^S$ and $RoA_c^S$ are not very significant. Thus, the idea to combine both Shuffle and Diversity may be useful to improve all those metrics. We formulate and combine Shuffle and Diversity technique to understand the benefits of deploying these techniques together on the cloud. To combine both Shuffle and Diversity techniques, we utilize VM-LM as the Shuffle technique and OS diversification as the Diversity techniques; then we deploy both VM-LM and OS diversification at the same time for each VM in the cloud. The strategies for ways combining MTD techniques can be different. For example, we can deploy Shuffle and Diversity on a single VM, or different VMs. In this section, we investigate the combination strategies and the effectiveness of each. Finding the best deployment strategy through Exhaustive Search (ES) is time-consuming on the large clouds. To address this problem, we use IMs to find out the most important VMs on the cloud first before applying MTD techniques on them.

We combine MTD techniques through considering the following strategies and steps after the first computation of IMs:

**$S+D$:** (*i*) deploying Shuffle on the VM indicated by IMs, (*ii*) deploying Diversity on the same VM;

**$D+S$:** (*i*) deploying Diversity on the VM indicated by IMs, (*ii*) deploying Shuffle on the same VM;

**$S\Delta D$:** (*i*) deploying Shuffle on the VM indicated by IMs, (*ii*) computing IMs again, (*iii*) deploying Diversity on a VM indicated by IMs;

**$D\Delta S$:** (*i*) deploying Diversity on the VM indicated by IMs, (*ii*) computing IMs again, (*iii*) deploying Shuffle on a VM indicated by IMs;

We define $S{\otimes}D$ as the combination of MTD techniques including different combination strategies. Although there are four possible combinations for combining Shuffle and Diversity using IMs, some combinations are equivalent based on the definition of HARM. For example, as Diversity only affects the lower layer of HARM, thus deploying Diversity first does not vary the upper layer of HARM. In this case, the permutation of $S{+}D$ or $D{+}S$ are the same and both techniques yield the same results. Similarly, the computation of IMs cannot be affected by deploying Diversity first. Then, the permutations of $S\Delta D$ or $D\Delta S$ are equivalent and both lead to the same results. The formal definitions of $S{\otimes}D$ include $S{+}D$ and $S\Delta D$ based on HARM are given as follows.

**Definition 7.** Let $S{\otimes}D(H, \kappa_s, \kappa_d)$ be a combination of Shuffle and Diversity function on the HARM where $1 \leq \kappa_s, \kappa_d \leq n$, and $\kappa_s$,$\kappa_d$ denote the specific VM $vm_{\kappa_s}$, $vm_{\kappa_d} \in VM$ chosen for the shuffling and OS diversification respectively. Then the result of $S{\otimes}D$ function is as $S{\otimes}D(H, \kappa_s, \kappa_d) = H^{\mathrm{s}\otimes\mathrm{d}}$. We define $H^{\mathrm{s}\otimes\mathrm{d}} = (U^{\mathrm{s},\kappa_s}, L^{\mathrm{d},k_d}, C)$ where $U^{\mathrm{s},\kappa_s}$ is equivalent to $U^{\mathrm{s},\kappa}$ in the Shuffle definition 5, where $\kappa = \kappa_s$, and $L^{\mathrm{d},\kappa_d}$ is equivalent to $L^{\mathrm{d},\kappa}$ in the Diversity definition 6, where $\kappa = \kappa_d$.

The $S{+}D$ can be a specific strategy under the definition of $S{\otimes}D(H, \kappa_s, \kappa_d)$, where $\kappa_s = \kappa_d$ which means deploying Shuffle and deploying Diversity on the same VM $vm_\kappa$ and named as $S{+}D(H, \kappa)$. The $S{+}D$ strategy includes two consecutive steps: $S{+}D(H, \kappa){=}S(H, \kappa){+}D(H^s, \kappa)$, which means first deploying Shuffle technique on the $vm_\kappa \in H$ where $\kappa$ is selected by $\kappa = \Gamma(H)$ and then deploy Diversity on the same VM $vm_\kappa$ in the transformed HARM $vm_\kappa \in H^s$.

Then, the $S\Delta D$ strategy can be defined as $S{\otimes}D(H, \kappa_s, \kappa_d)$, where $\kappa_s$ and $\kappa_d$ are not necessarily the same and is denoted as $S\Delta D(H, \kappa_s, \kappa_d)$. The $S\Delta D$ includes two consecutive steps: $S\Delta D(H, \kappa_s, \kappa_d) = S(H, \kappa_s) + D(H^s, \kappa_d)$, which specifically means first deploying Shuffle technique on the VM selected by $\kappa_s{=}\Gamma(H)$ which is $vm_{\kappa_s}$. Then deploy Diversity on the VM chosen by $\kappa_d = \Gamma(H^s)$ which is $vm_{\kappa_d}{\in}H^s$ which exists on the transformed HARM resulting from deploying Shuffle.

### *5.1. Evaluation of MTD Combinations*

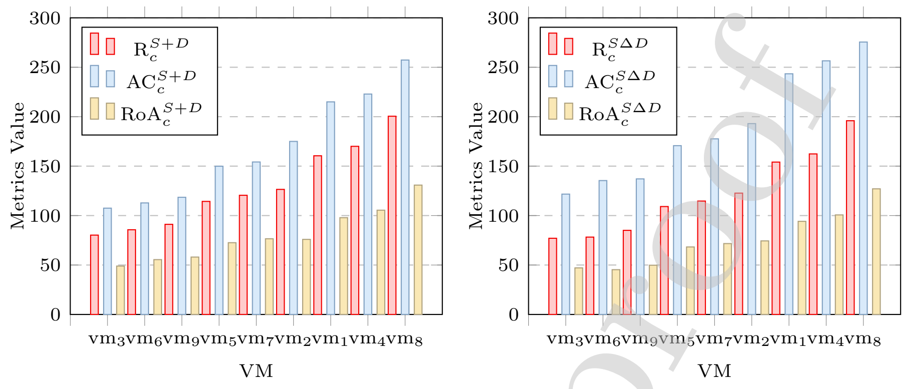
In this section, we evaluate the effectiveness of combinations of Shuffle and Diversity MTD techniques. This evaluation undergoes three steps: (1) creating the cloud model using HARM, (2) deploying different combination strategies (based on $S{+}D$ and $S{\otimes}D$), and finally (3) analyzing the changes in the cloud's security posture for each strategy.

We analyze the security metrics values after deploying each $S{\otimes}D$ strategies. The following example shows the deploying $S{+}D$ on the cloud based on the given Definition 7.

**Example 5.** The result of $S{+}D$ function on the generated HARM for the cloud example ($H_{cs}$) in which a VM is selected to be shuffled ($vm_{\kappa}$) can be represented as $S{+}D(H_{cs}, \kappa) = S(H_{cs}, \kappa) + D(H^s_{cs}, \kappa)$ including the following steps: ($i$) Finding the most important VM in the HARM using $\kappa = \Gamma(H)$, which returns $\kappa = 3$, ($ii$) Deploying $S(H_{cs}, 3)$, where $vm_3$ is shuffled and results in the changes on the upper layer of HARM ($H^s$). ($iii$) Deploying Diversity on the transformed HARM $D(H^s_{cs}, 3)$, where the selected VM for deploying Diversity is the same as the previous step which was $vm_3$.

We denote the security metrics resulting from deploying $S{+}D$ as $R_c^{S+D}$, $AC_c^{S+D}$, and $RoA_c^{S+D}$. Similarly, we can show the example of deploying $S\Delta D$ as follows.

**Example 6.** The result of $S\Delta D$ function on the generated HARM for the cloud example ($H_{cs}$) in which two VMs are selected for deploying combined MTD can be represented as $S\Delta D(H_{cs}, \kappa_s, \kappa_d) = S(H_{cs}, \kappa_s) + D(H^s_{cs}, \kappa_d)$ consisting of the following steps: ($i$) Finding the most important VM in the HARM using $\kappa_s = \Gamma(H)$, which returns $\kappa_s = 3$ meaning $vm_3$ is selected, ($ii$) Deploying $S(H_{cs}, 3)$, where $vm_3$ is shuffled and results in the changes on the HARM ($H^s$). ($iii$) Re-calculating IMs again on the transformed HARM ($H^s_{cs}$) to find the most important VM for deploying Diversity using $\kappa_d = \Gamma(H^s_{cs})$, which returns $\kappa_d = 6$ meaning $vm_6$ is selected, ($iv$) Deploying Diversity on the transformed HARM $D(H^s_{cs}, 6)$, where the selected VM for deploying Diversity is $vm_6$.



(a) $S{+}D$ (b) $S\Delta D$

Figure 6: Comparison of security metrics after deploying two MTD combination strategies on the VMs (VMs in x-axis are sorted in descending order based on their importance using IMs).

We denote the security metrics resulting from deploying $S\Delta D$ as $R_c^{S\Delta D}$, $AC_c^{S\Delta D}$, and $RoA_c^{S\Delta D}$. We consider two combination strategies of MTD and evaluate the cloud security level before and after deploying the combined techniques. Then, we compare the combination strategies to find a more effective technique. Figure 6 compares the results of deploying two different combinations of MTD techniques which are $S{+}D$ and $S\Delta D$ on the VM in the cloud example through analysis of the security metrics. The metric values are plotted as the y-axis against the VMs sorted in the descending trend based on their IMs values in the x-axis. Figure 6a shows the security metrics after deploying $S{+}D$, and Figure 6b reveals the results of deploying $S\Delta D$ on the cloud example. Both graphs show the combinations of MTD techniques on VMs which have higher IM values (like $vm_3$, $vm_6$, and $vm_9$) cause lower security metrics rates. Although deploying the combined MTD decreases $AC_c$ values, it also intensively reduces $R_c$, and $RoA_c$ values. With these results, we can observe that it is important to choose the appropriate VM. For instance, deploying MTD techniques on the VMs with the lower values of IMs are shown to be less effective than the others. Moreover, we can observe that $S\Delta D$ deployment strategy is better than $S{+}D$ as the $R_c^{S\Delta D}$ and $RoA_c^{S\Delta D}$ are lower than $R_c^{S+D}$ and $RoA_c^{S+D}$, while $AC_c^{S\Delta D}$ rates are higher than $AC_c^{S+D}$ values. However, the obtained security

Table 6: The percentage of changes in the security metrics after deploying S+D on each VM in the cloud example

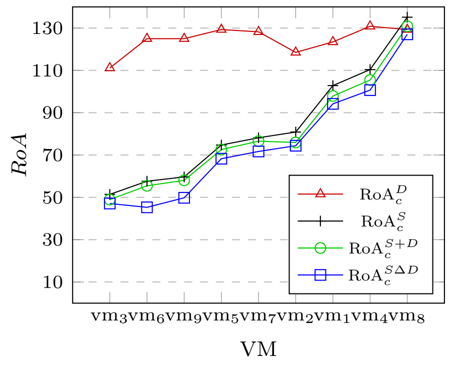
| VM | Values | | | % of Changes | | |
|---|---|---|---|---|---|---|
| | $R_c^{S+D}$ | $AC_c^{S+D}$ | $RoA_c^{S+D}$ | $R_c^{S+D}$ | $AC_c^{S+D}$ | $RoA_c^{S+D}$ |
| vm$_1$ | 160.48 | 215 | 97.87 | -24.19% | -14.48% | -27.91% |
| vm$_2$ | 126.49 | 175 | 75.93 | -40.25% | -30.39% | -44.07% |
| vm$_3$ | **80.24** | 107.5 | **48.94** | **-62.10%** | -57.24% | **-63.95%** |
| vm$_4$ | 170.03 | 222.9 | 105.44 | -19.68% | -11.34% | -22.33% |
| vm$_5$ | 114.34 | 149.9 | 72.58 | -45.99% | -40.37% | -46.54% |
| vm$_6$ | 85.66 | 112.8 | 55.46 | -59.53% | -55.13% | -59.15% |
| vm$_7$ | 120.53 | 154.2 | 76.54 | -43.06% | -38.66% | -43.62% |
| vm$_8$ | 200.47 | 257.3 | 130.82 | -5.30% | 2.35% | -3.64% |
| vm$_9$ | 91.15 | 118.5 | 58.03 | -56.94% | -52.86% | -57.25% |

metrics values depend on the cloud's constraints and might be different in other contexts. Based on the results, the optimal results of deploying $S{+}D$ and $S\Delta D$ are the selection of $vm_3$ leading to the best values.

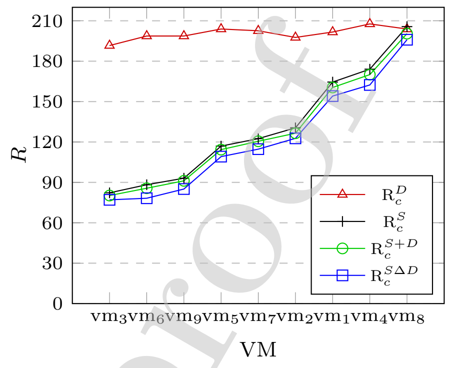
Comparing the percentage of the changes on metrics reported in Table 6 and 7 for combinations of MTD techniques with Table 5 for $D$ only, we can notice a significant improvement on $R_c^D$ and $RoA_c^D$, where the percentage of the changes for those values are about **-9.49** and **-18.16** respectively in the best scenario (deploying Diversity on $vm_3$), while these rates for $R_c^{S+D}$ and $RoA_c^{S+D}$ are about **-62.1%** and **-63.9%** for the best deployment scenario, and even better for $S\Delta D$ with the values of **-63.6%** and **-65.3%** for $R_c^{S\Delta D}$ and

Table 7: The percentage of changes in the security metrics after deploying S∆D on each VM in the cloud example

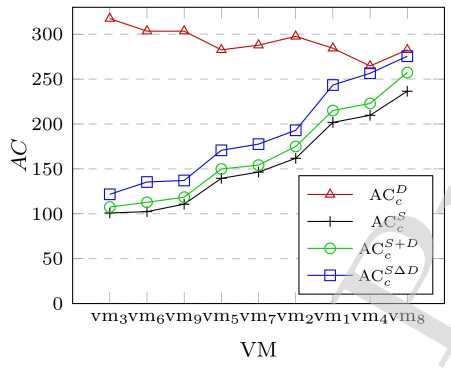
| VM | Values | | | % of Changes | | |
|---|---|---|---|---|---|---|
| | $R_c^{S\Delta D}$ | $AC_c^{S\Delta D}$ | $RoA_c^{S\Delta D}$ | $R_c^{S\Delta D}$ | $AC_c^{S\Delta D}$ | $RoA_c^{S\Delta D}$ |
| vm$_1$ | 154.09 | 243.4 | 94.18 | -27.21% | -3.18% | -30.63% |
| vm$_2$ | 122.71 | 193 | 74.39 | -42.03% | -23.23% | -45.20% |
| vm$_3$ | **77.05** | 121.7 | **47.09** | **-63.60%** | -51.59% | **-65.31%** |
| vm$_4$ | 162.35 | 256.5 | 100.67 | -23.31% | 2.03% | -25.85% |
| vm$_5$ | 109.14 | 170.7 | 68.26 | -48.44% | -32.10% | -49.72% |
| vm$_6$ | 78.22 | 135.4 | 45.29 | -63.05% | -46.14% | **-66.65%** |
| vm$_7$ | 114.68 | 177.6 | 71.69 | -45.83% | -29.36% | -47.19% |
| vm$_8$ | 195.92 | 275.5 | 127.04 | -7.45% | 9.59% | -6.42% |
| vm$_9$ | 85.07 | 137.1 | 49.79 | -59.81% | -45.47% | -63.32% |



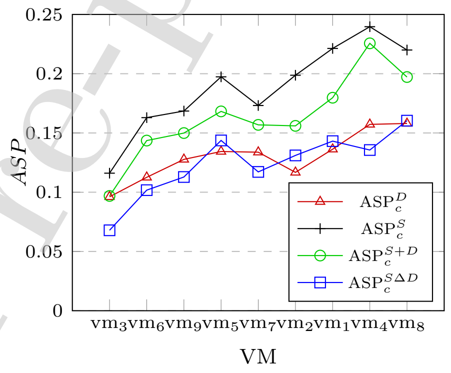
(a) $RoA_c$ values



(b) $R_c$ values



(c) $AC_c$ values



(d) $ASP_c$ values

Figure 7: Comparison of security metrics resulting from deploying various MTD techniques (VMs in x-axis are sorted in descending order based on their importance using IMs).

$RoA_c^{S\Delta D}$ respectively.

Figure 7 compares the results of deploying each MTD technique ($S$ and $D$) and combination strategies ($S\Delta D$ and $S{+}D$) on the security metrics ($RoA_c$, $R_c$, $AC_c$, and $ASP_c$). The results demonstrated in 7a show that $RoA_c^{S\Delta D}$ has lower values than the other MTD techniques and combination strategies for all selected VMs. However, the best $RoA_c$ results can be found through deploying $S\Delta D$ on the top three VMs $vm_3$, $vm_6$, and $vm_9$ having the higher IMs rates which yield $RoA_c^{S\Delta D}$ lower than 50. Nevertheless, the worst results for $RoA_c$ can be obtained after deploying Diversity which yields the highest $RoA_c$ as $RoA_c^D$ values fall between **110** and **130** in the best and the worst deployment scenarios (shown in Figure 7a). The same trend can be seen in the results of deploying

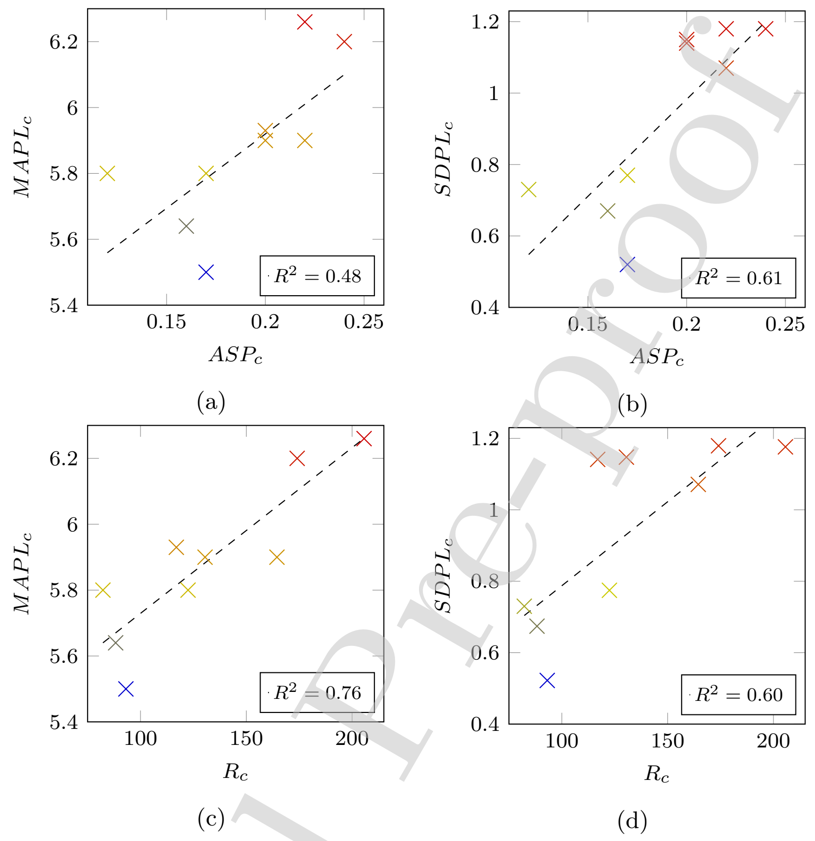


Figure 8: Comparing the correlation of path-based metrics against $ASP_c$ and $R_c$ after deploying Shuffle on each VM. The correlation co-efficient between (a) $ASP_c$ and $MAPL_c$ is about 70%, (b)$ASP_c$ and $SDPL_c$ is about 78%, (c)$R_c$ and $MAPL_c$ is about 87%, (d)$ASP_c$ and $SDPL_c$ is about 77% showing a positive relation.

MTD strategies on $R_c$ values. The best $R_c$ results from deploying $S\Delta D$ as $R_c^{S\Delta D}$ for all VMs have lower values than the $R_c$ obtained from the other MTD strategies. In contrast, $R_c^D$ yield the worst results, see Figure 7b. The results of $AC_c$ are shown in Figure 7c. Comparing the $AC_c$ values obtained from deploying different MTD strategies, we observe that $AC_c^D$ values are highest and $AC_c^S$ values are the lowest among the other techniques. Figure 7d compares all $ASP_c$ values resulting from deploying different MTD strategies on the VMs. As it can be seen, the results of $ASP_c^{S\Delta D}$ yield the lower $ASP_c$ values in comparison with the other MTD strategies. It means that after deploying $S\Delta D$ the attacker has a lower chance to compromise the target while this chance after deploying other MTD strategies are higher. The worst $ASP_c$ values can be found after deploying Shuffle as the $ASP_c^S$ are the highest. Furthermore, in all cases, deploying MTD strategies on the VMs having higher IMs values leads to better results in terms of the security metrics ($R_c$, $RoA_c$, $AC_c$, and $ASP_c$). Moreover, considering the overall results, we can conclude that combining MTD techniques leads to better results in terms of the security metrics. We also observe that the combination strategies are important to achieve a better security posture as the results of $S\Delta D$ surpasses $S{+}D$ values.

However, deploying Diversity has no effect on the reachability of the VMs, whilst deploying Shuffle varies the reachability of VMs and changes the upper layer of HARM. Thus, Shuffle may change the path-based metrics. Moreover, as Diversity technique cannot affect the upper layer of HARM, it cannot vary the path-based metrics. Consequently, the results of deploying those MTD techniques combined with Shuffle ($S$, $S{+}D$, and $S\Delta D$) on the path-based metrics are similar. Thus, we only evaluate the path-based metrics for Shuffle technique. We computed the path-based metrics after deploying Shuffle on each VM in the cloud. In Figure 8, we plot the $ASP$ and $R$ values obtained after deploying Shuffle on each VM against the corresponding values for $MAPL$ and $SDPL$ and compute the correlation coefficient of those metrics. We found a positive linear correlation between them which means increment on path-based metrics like $MAPL$ and $SDPL$ may increase the risk and attack success probability. Figure 9 shows the normalized values of path-based metrics and the changes in those metrics after deploying Shuffle on each VM. The values of x-axis are the VMs sorted in descending order based on their importance using IMs (as Equation 1). The results demonstrate a gentle increment on the both $MAPL$ and $SDPL$ values. As the higher values of $MAPL$ and $SDPL$ may increase the $ASP$ and $R$ values, deploying Shuffle on the VMs $vm_9$, $vm_6$, and $vm_3$ which have higher betweenness values leads better $MAPL$ and $SDPL$ values. Moreover, although the $SAP$ values remain steady after deploying Shuffle on each VM, deploying Shuffle on VMs $vm_3$, $vm_9$, and $vm_8$ have the higher $SAP$

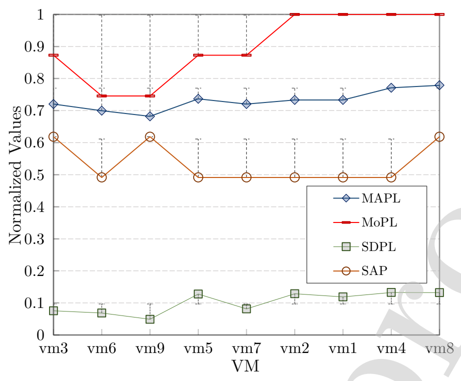


Figure 9: Changes on the Path-based Metrics after deploying Shuffle on each VM (VMs in x-axis are sorted in descending order based on their importance using IMs).

values showing the minimum number of VMs that the attacker needs to exploit to compromise the target are higher for those VMs.

## *5.2. Simulation and Evaluation in Large Cloud Model*

We assess the effectiveness of individual and combined MTD techniques in a larger context. The evaluation in the larger scale can show that (1) how scalable the proposed MTD techniques and strategies are, (2) how the cloud security posture varies as the size of the cloud changes (as the management of the larger clouds is more difficult than smaller clouds). We simulated a large Cloud-band model we used in the earlier version of this paper [18]. This cloud-band model includes up to 900 VMs divided into two cloud-band nodes each of which can accommodate up to 450 VMs. We assume that only a few VMs are the entry points of the system (i.e., front-end servers) and are connected to the Internet. Moreover, we assume that the attackers can only enter the cloud from the VMs connected to the internet. Attackers can exploit vulnerabilities on each VM, explore the attack paths, and finally get access to the resource node. We also assume that all VMs in the cloud-band nodes use the Linux as the main OS and Fedora as the backup OS. The vulnerability values for each OS are considered based on the values in Table 1. We also assume that the backup OS for each VM in the cloud can be launched by the cloud provider with the neglectable machine downtime. Finally, the cloud provider allows VM-LM feature between cloud-band nodes as if there is available space on each node. Note that VM-LM may rearrange the logical connectivities between the VMs based on the cloud constraints.

In Section 4, we showed that deploying MTD techniques on the VMs with higher IM values leads to better security posture in comparison with other VMs. To deploy MTD techniques effectively, we used IMs rather than ES. We deploy MTD techniques in the cloud-band example and consider the effectiveness of each MTD combination by comparing the security metrics. We first generate the two layered-HARM for the cloud-band model and analyze the security posture of the cloud before and after deploying each MTD technique.

To conduct the results, we simulate the various cloud-band sizes and deploy for deploying MTD techniques. Figure 10 compares the security posture of the cloud with various VM sizes before and after deploying the MTD techniques: $S$, $D$, $S$+$D$, and $S\Delta D$. Considering the graphs, we can obvious that deploying MTD decreases $R$ and $RoA$ values for all techniques. Moreover, $AC$ increases in $D$, while this value decreases in the other techniques. However, in $D$ technique, as in Figure 10b, the decrements in $R$ and $RoA$ rates are less in comparison with the others which infer that the other techniques have better results than $D$ based on the defined cloud and system constraints. Then, the results show that using $S$ in MTD can be helpful for cloud providers as it highly decreases the values of $R$ and $RoA$, as in Figure 10a. We also can see that increasing the cloud-band sizes provides the linear increment on all metrics. This trend is almost similar in $S$ and $S$+$D$, but different in $D$ and $S\Delta D$. For instance, the overall values for the metrics are very high in $D$ only. Although it increases the $AC$ values, the values of $RoA$ after deploying $D$ are very high. For instance, this value is about 777 for the cloud-band with 400 VMs which shows the high orientation of the attacker to attack the cloud again, while the same $RoA$ values are about halved for other techniques.

**The best scenario.** Although Diversity ($D$) provides the worst results in $R$ and $RoA$ comparing with the other techniques, it increases the $AC$ rates. Thus, combining $D$ with $S$ improves the overall security posture of the cloud.

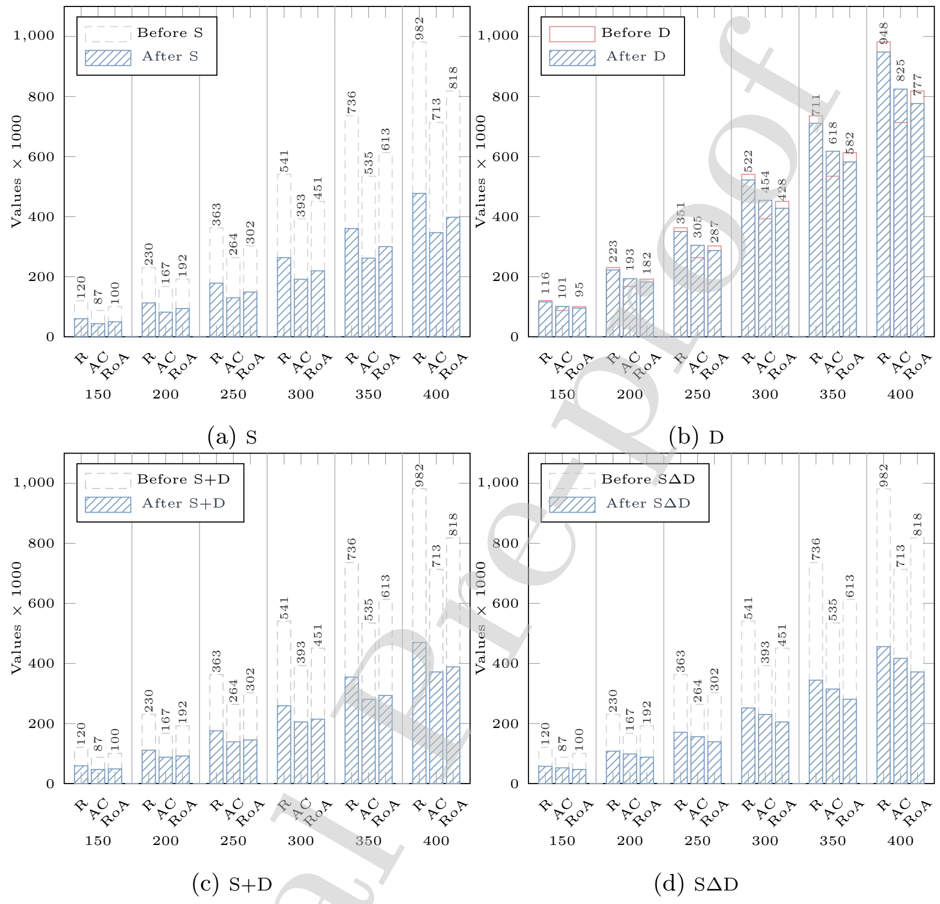


Figure 10: Comparison of metrics before and after deploying the four types of MTD metrics.

When the IMs are calculated between the techniques $S\Delta D$, it leads to the best combination scenario, see Figure 10d. Moreover, the security posture of $S$+$D$ is better than $S$ or $D$-only, see Figure 10c, but lower than $S\Delta D$. Comparing the results, we observe that the best MTD combination is $S\Delta D$ technique in which the values of $R$ and $RoA$ are lower than the others, the $RoA$ values in $S\Delta D$ are 47 and 372 for the cloud-band with 150 and 400 VMs respectively, which shows the optimal values. Moreover, the optimal values for $R$ are 57 and 456 for 150 and 400 VMs in the cloud-bands respectively. Furthermore, the values for $AC$ when $S\Delta D$ are deployed are better than those of $S$ and $S$+$D$ techniques.

## 6. Discussion and Limitations

In this paper, we proposed the novel combinations of MTD techniques which utilize different strategies to combine Shuffle and Diversity techniques. The four MTD deployment scenarios include Shuffle and Diversity as the single MTD techniques together with $S{+}D$ and $S\Delta D$ as the combined strategies. The approaches we used to deploy the MTD techniques on the cloud are limited to VM-LM and OS diversification for Shuffle and Diversity respectively. We utilized four main security metrics $R$, $AC$, $RoA$, and $ASP$ to evaluate the security posture of the cloud. We chose to use those metrics due to evaluating the cloud security posture based on two different perspectives. Two $AC$ and $RoA$ show the security level of the cloud based on the attacker's perspective, while $R$ and $ASP$ are categorized as the metrics showing the cloud security based on the cloud providers' perspective. Then, we evaluated the results of deploying MTD techniques on the path-based metrics to observe the cloud security based on the reachability of the VMs. Path-based metrics can show the attacker's efforts needed to compromise the target in terms of exploiting the paths, the number of VMs on a path, minimum path length, and so on. We further discuss the limitations and extensions of this work.

### *6.1. Combining MTD Techniques*

A comparison of current MTD literature which proposed the combination MTD technique is presented in Table 8. This comparison is based on factors such as MTD techniques, combination strategies (as discussed in Section 5), and security metrics including path-based metrics. We investigated the combining MTD techniques as each technique can be effective on a particular sight. For instance, Diversity incurs more costs for the attacker and can be evaluated in terms of the attacker's point of view. Some security metrics such as $AC$ and $RoA$ are useful indicators to quantify the security of the cloud based on the attacker's perspective. We showed how Diversity increases $AC$ (shown in Figure 10b). In fact, the main drawback of Diversity is that it may need extra

Table 8: Comparison of the MTD papers with regards to combinations of MTD, strategies, and security metrics

| Ref. | MTD Techniques | Combination Strategies | Security Metrics | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | R | ASP | Rel | AC | RoA | MAPL | MoPL | SDPL | ASP |
| This paper | S, D | ✓ | ✓ | ✓ | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| [18] | S, R, D | ✗ | ✓ | ✗ | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ |
| [11] | S, R | ✗ | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| [19] | S, D | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ |

resources (such as more Operating Systems (OS) variants) and lack of enough variants may affect the Diversity functionality. Alternatively, Shuffle may be deployed in such a way to improve the overall security of the system such as reducing the Risk and RoA values. The inherent benefit of Shuffle is that it can be easily applicable and usually no need extra resources. However, as Shuffle relies on the existing resources, it cannot eliminate or change the vulnerabilities of the current resources. Consequently, those MTD techniques can be well-mingled together by the cloud provider to benefit from the advantages of each technique.

### *6.2. Limitations and Extensions*

In this paper, we only considered Shuffle, Diversity and combination of those two techniques, while the other combinations of MTD techniques can also be formalized and investigated. Although analyzing more MTD techniques and combinations of them might increase the security analysis complexity, it might provide the cloud providers with better security solutions and defensive capabilities and the cloud provider can opt the desirable MTD techniques for combination based on the available resources.

Despite their importance, the security metrics we used in our experiments are only limited to $R$, $AC$, $RoA$, and $ASP$ together with path-based metrics while there are other security metrics which can be used based on the cloud security level requirements, such as those discussed in [57]. We will further consider other sets of security metrics to capture different aspects of the cloud security requirements in our future work. Moreover, in this paper, we assumed that the attacker is outside of the cloud. However, adapting an MTD technique, which is able to defend against the insider threats, is also important. The techniques to identify the propagation source of attacks in a network can be utilized to enhance the defensive techniques against the prospective attack from a single or a group of suspicious sources [58, 59]. Moreover, the techniques for detecting and preventing attacks launched from inside of a network are extensively discussed in [60]. We believe that our MTD techniques can be expanded based on intrusion alerts, then the MTD triggers when a threat is detected by the Intrusion Detection System (IDS).

We considered and evaluated Diversity technique with only a single backup VM. Thus, the security achieved by the Diversity technique can be significantly degraded by the limitation on the number of variants in such a way that the attacker may gradually learn the techniques to exploit and cover a set of limited vulnerabilities. To address this, we further investigate Diversity technique with more backup OS variants. Moreover, other Diversity approaches such as Applications, Code and Software Diversity [30] can be included.

Moreover, there is a lack of the economical evaluation of deploying MTD techniques on the cloud. For example, in order to deploy Diversity technique the cloud provider should purchase more OS or licenses. It is difficult for most of the private cloud providers to supply various OS variants for their customers due to economical-related reasons; vendors can provide the customers with more OS options, but customers may not afford to buy them. Thus, cost evaluation of deploying MTD techniques is a crucial problem which we plan to address in our future work.

## 7. Conclusion

MTD techniques can be applied to cloud computing to enhance the security of the cloud by making the cloud more unpredictable for the attackers. In this paper, we introduced the combinations of MTD techniques including Shuffle and Diversity and evaluated the effectiveness of them by deploying different combination strategies for the cloud. Comparing the security posture of the cloud before and after deploying each combination scenario, we showed that combining MTD techniques is important, as it can introduce additional benefits of enhancing security, which may not be possible under a single MTD technique. For instance, Diversity technique can enhance some security metrics such as $AC$ and $RoA$ which are mainly used to show the cloud security level based on the attacker's perspective, while Shuffle technique may vary other aspects of security metric like $R$ and $ASP$ which mainly capture the security from cloud providers' point of view. Thus, MTD techniques can be well-mingled together aiming to increase various aspects of security. We also wanted to show that, if those techniques are not properly deployed, it may cost the cloud providers and also it may increase the attack surface. Moreover, we simulated a large cloud-band model to conduct the results. Our experimental analysis showed the best combination strategies for Shuffle and Diversity providing better results in terms of improving the security posture of the cloud in comparison to single MTD techniques or other combinations.

## References

[1] K. Popović, Ž. Hocenski, Cloud computing security issues and challenges, in: The 33rd International Convention MIPRO, IEEE, 2010, pp. 344–349.

[2] Y. Zhu, H. Hu, G. Ahn, D. Huang, S. Wang, Towards temporal access control in cloud computing, in: Proc. of Annual IEEE International Conference on Computer Communications (INFOCOM 2012), 2012, pp. 2576–2580.

[3] Q. Jia, H. Wang, D. Fleck, F. Li, A. Stavrou, W. Powell, Catch Me if You Can: A Cloud-Enabled DDoS Defense, in: Proc. of the the 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2014), 2014.

[4] J. B. Hong, D. S. Kim, Assessing the effectiveness of moving target

defenses using security models, IEEE Transactions on Dependable and Secure Computing 13 (2) (2016) 163–177.

[5] J. H. Jafarian, E. Al-Shaer, Q. Duan, Openflow random host mutation: transparent moving target defense using software defined networking, in: Proceedings of the first workshop on Hot topics in software defined networks, ACM, 2012, pp. 127–132.

[6] S. Vikram, C. Yang, G. Gu, Nomad: Towards non-intrusive moving-target defense against web bots, in: Communications and Network Security (CNS), 2013 IEEE Conference on, IEEE, 2013, pp. 55–63.

[7] E. Yuan, S. Malek, B. Schmerl, D. Garlan, J. Gennari, Architecture-based self-protecting software systems, in: Proceedings of the 9th international ACM Sigsoft conference on Quality of software architectures, ACM, 2013, pp. 33–42.

[8] J. P. Rohrer, A. Jabbar, J. P. Sterbenz, Path diversification for future internet end-to-end resilience and survivability, Telecommunication Systems 56 (1) (2014) 49–67.

[9] A. Newell, D. Obenshain, T. Tantillo, C. Nita-Rotaru, Y. Amir, Increasing network resiliency by optimally assigning diverse variants to routing nodes, IEEE Transactions on Dependable and Secure Computing 12 (6) (2015) 602–614.

[10] M. A. Khan, K. Salah, Iot security: Review, blockchain solutions, and open challenges, Future Generation Computer Systems 82 (2018) 395–411.

[11] H. Alavizadeh, D. S. Kim, J. B. Hong, J. Jang-Jaccard, Effective security analysis for combinations of mtd techniques on cloud computing (short paper), in: International Conference on Information Security Practice and Experience, Springer, 2017, pp. 539–548.

[12] M. Ge, J. B. Hong, S. E. Yusuf, D. S. Kim, Proactive defense mechanisms for the software-defined internet of things with non-patchable vulnerabilities, Future Generation Computer Systems 78 (2018) 568–582.

[13] A. M. Nhlabatsi, J. B. Hong, D. S. D. Kim, R. Fernandez, A. Hussein, N. Fetais, K. M. Khan, Threat-specific security risk evaluation in the cloud, IEEE Transactions on Cloud Computing.

[14] J. Hong, D.-S. Kim, Harms: Hierarchical attack representation models for network security analysis.

[15] J. B. Hong, D. S. Kim, Performance analysis of scalable attack representation models, in: IFIP International Information Security Conference, Springer, 2013, pp. 330–343.

[16] P. K. Manadhata, Game theoretic approaches to attack surface shifting, in: Moving Target Defense II, Springer, 2013, pp. 1–13.

[17] H.-q. Zhang, C. Lei, D.-x. Chang, Y.-j. Yang, Network moving target defence technique based on collaborative mutation, Computers & Security.

[18] H. Alavizadeh, J. B. Hong, J. Jang-Jaccard, D. S. Kim, Comprehensive security assessment of combined mtd techniques for the cloud, in: Proceedings of the 5th ACM Workshop on Moving Target Defense, ACM, 2018, pp. 11–20.

[19] H. Alavizadeh, J. Jang-Jaccard, D. S. Kim, Evaluation for combination of shuffle and diversity on moving target defense strategy for cloud computing, in: 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), IEEE, 2018, pp. 573–578.

[20] F. T. Sheldon, C. Vishik, Moving toward trustworthy systems: R&d essentials, Computer 43 (9) (2010) 31–40.

[21] R. Zhuang, S. Zhang, A. Bardas, S. DeLoach, X. Ou, A. Singhal, Investigating the Application of Moving Target Defenses to Network Security, in: Proc. of the 6th International Symposium on Resilient Control Systems (ISRCS 2013), 2013, pp. 162–169. doi:10.1109/ISRCS.2013.6623770.

[22] Y. Zhang, M. Li, K. Bai, M. Yu, W. Zang, Incentive compatible moving target defense against vm-colocation attacks in clouds., in: SEC, Springer, 2012, pp. 388–399.

[23] S. Venkatesan, M. Albanese, K. Amin, S. Jajodia, M. Wright, A moving target defense approach to mitigate ddos attacks against proxy-based architectures, in: Communications and Network Security (CNS), 2016 IEEE Conference on, IEEE, 2016, pp. 198–206.

[24] B. Chatfield, R. J. Haddad, Moving target defense intrusion detection system for ipv6 based smart grid advanced metering infrastructure, in: SoutheastCon, 2017, IEEE, 2017, pp. 1–7.

[25] E. Al-Shaer, Toward network configuration randomization for moving target defense, Moving Target Defense (2011) 153–159.

[26] L. Zhang, S. Shetty, P. Liu, J. Jing, Rootkitdet: Practical end-to-end defense against kernel rootkits in a cloud environment, in: European Symposium on Research in Computer Security, Springer, 2014, pp. 475–493.

[27] F. Al-Haidari, M. Sqalli, K. Salah, Impact of cpu utilization thresholds and scaling size on autoscaling cloud resources, in: 2013 IEEE 5th International Conference on Cloud Computing Technology and Science, Vol. 2, IEEE, 2013, pp. 256–261.

[28] P. Calyam, S. Rajagopalan, S. Seetharam, A. Selvadhurai, K. Salah, R. Ramnath, Vdc-analyst: Design and verification of virtual desktop cloud resource allocations, Computer Networks 68 (2014) 110–122.

[29] S. Antonatos, P. Akritidis, E. P. Markatos, K. G. Anagnostakis, Defending against hitlist worms using network address space randomization, Computer Networks 51 (12) (2007) 3471–3490.

[30] Y. Huang, A. Ghosh, Introducing Diversity and Uncertainty to Create Moving Attack Surfaces for Web Services, in: S. Jajodia, A. K. Ghosh, V. Swarup, C. Wang, X. S. Wang (Eds.), Moving Target Defense, Vol. 54 of Advances in Information Security, Springer New York, 2011, pp. 131–151.

[31] M. Azab, R. Hassan, M. Eltoweissy, Chameleonsoft: a moving target defense system, in: Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), 2011 7th International Conference on, IEEE, 2011, pp. 241–250.

[32] J. Steinberger, B. Kuhnert, C. Dietz, L. Ball, A. Sperotto, H. Baier, A. Pras, G. Dreo, Ddos defense using mtd and sdn, in: NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium, IEEE, 2018, pp. 1–9.

[33] N. Z. Bawany, J. A. Shamsi, K. Salah, Ddos attack detection and mitigation using sdn: methods, practices, and solutions, Arabian Journal for Science and Engineering 42 (2) (2017) 425–441.

[34] B. Tozer, T. Mazzuchi, S. Sarkani, Optimizing attack surface and configuration diversity using multi-objective reinforcement learning, in: Proceedings of the IEEE 14th International Conference on Machine Learning and Applications (ICMLA), 2015, pp. 144–149.

[35] N. Sun, J. Zhang, P. Rimba, S. Gao, L. Y. Zhang, Y. Xiang, Data-driven cybersecurity incident prediction: A survey, IEEE Communications Surveys & Tutorials 21 (2) (2018) 1744–1772.

[36] S. Sengupta, A. Chowdhary, D. Huang, S. Kambhampati, Moving target defense for the placement of intrusion detection systems in the cloud,

in: International Conference on Decision and Game Theory for Security, Springer, 2018, pp. 326–345.

[37] J. Zhang, Y. Xiang, Y. Wang, W. Zhou, Y. Xiang, Y. Guan, Network traffic classification using correlation information, IEEE Transactions on Parallel and Distributed systems 24 (1) (2012) 104–117.

[38] T. Ristenpart, E. Tromer, H. Shacham, S. 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, ACM, 2009, pp. 199–212.

[39] B. Danev, R. J. Masti, G. O. Karame, S. Capkun, Enabling secure vm-vtpm migration in private clouds, in: Proceedings of the 27th Annual Computer Security Applications Conference, ACM, 2011, pp. 187–196.

[40] T. Penner, M. Guirguis, Combating the bandits in the cloud: A moving target defense approach, in: Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, IEEE Press, 2017, pp. 411–420.

[41] W. Peng, F. Li, C.-T. Huang, X. Zou, A moving-target defense strategy for cloud-based services with heterogeneous and dynamic attack surfaces, in: Communications (ICC), 2014 IEEE International Conference on, IEEE, 2014, pp. 804–809.

[42] P. Mell, K. Scarfone, S. Romanosky, Common vulnerability scoring system, IEEE Security & Privacy 4 (6).

[43] D. P. Sharma, D. S. Kim, S. Yoon, H. Lim, J.-H. Cho, T. J. Moore, Frvm: Flexible random virtual ip multiplexing in software-defined networks, in: 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), IEEE, 2018, pp. 579–587.

[44] Unitecloud, http://www.unitecloud.net/.

[45] H. Alavizadeh, H. Alavizadeh, D. S. Kim, J. Jang-Jaccard, M. N. Torshiz, An automated security analysis framework and implementation for cloud, arXiv preprint arXiv:1904.01758.

[46] S. Y. Enoch, M. Ge, J. B. Hong, H. Alzaid, D. S. Kim, A systematic evaluation of cybersecurity metrics for dynamic networks, Computer Networks 144 (2018) 216–229.

[47] S. E. Yusuf, M. Ge, J. B. Hong, H. K. Kim, P. Kim, D. S. Kim, Security modelling and analysis of dynamic enterprise networks, in: Computer and Information Technology (CIT), 2016 IEEE International Conference on, IEEE, 2016, pp. 249–256.

[48] F. Cadini, E. Zio, C.-A. Petrescu, Using centrality measures to rank the importance of the components of a complex network infrastructure., in: CRITIS, Springer, 2008, pp. 155–167.

[49] M. Cremonini, P. Martini, Evaluating information security investments from attackers perspective: the return-on-attack (roa), in: WEIS, 2005.

[50] S. E. Yusuf, J. B. Hong, M. Ge, D. S. Kim, Composite metrics for network security analysis, Software Networking 2018 (1) (2018) 137–160.

[51] A. Roy, D. Kim, K. Trivedi, Attack Countermeasure Trees (ACT): Towards Unifying the Constructs of Attack and Defense Trees, Security and Communication Networks 5 (8) (2012) 929–943.

[52] R. A. Sahner, K. Trivedi, A. Puliafito, Performance and Reliability Analysis of Computer Systems: An Example-Based Approach Using the SHARPE Software Package, Springer Publishing Company, Incorporated, 2012.

[53] M. T. Adili, A. Mohammadi, M. H. Manshaei, M. A. Rahman, A cost-effective security management for clouds: A game-theoretic deception

mechanism, in: Integrated Network and Service Management (IM), 2017 IFIP/IEEE Symposium on, IEEE, 2017, pp. 98–106.

[54] G.-l. Cai, B.-s. Wang, W. Hu, T.-z. Wang, Moving target defense: state of the art and characteristics, Frontiers of Information Technology & Electronic Engineering 17 (11) (2016) 1122–1153.

[55] J. Rohrer, A. Jabbar, J. Sterbenz, Path Diversification for Future Internet End-to-End Resilience and Survivability, Telecommunication Systems (2013) 1–19doi:10.1007/s11235-013-9818-7.

[56] J. Hong, D. Kim, A. Haqiq, What Vulnerability Do We Need to Patch First?, in: Proc. of the 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshop (DSNW 2014), 2014, pp. 684–689. doi:10.1109/DSN.2014.68.

[57] M. Pendleton, R. Garcia-Lebron, J.-H. Cho, S. Xu, A survey on systems security metrics, ACM Computing Surveys (CSUR) 49 (4) (2017) 62.

[58] J. Jiang, S. Wen, S. Yu, Y. Xiang, W. Zhou, Identifying propagation sources in networks: State-of-the-art and comparative studies, IEEE Communications Surveys & Tutorials 19 (1) (2016) 465–481.

[59] S. Wen, M. S. Haghighi, C. Chen, Y. Xiang, W. Zhou, W. Jia, A sword with two edges: Propagation studies on both positive and negative information in online social networks, IEEE Transactions on Computers 64 (3) (2014) 640–653.

[60] L. Liu, O. De Vel, Q.-L. Han, J. Zhang, Y. Xiang, Detecting and preventing cyber insider threats: a survey, IEEE Communications Surveys & Tutorials 20 (2) (2018) 1397–1417.

**Hooman Alavizadeh** is a PhD candidate in School of Natural and Computational Sciences, Massey University, Auckland, New Zealand. He received his M.Sc. Degree in Computer Science from Eastern Mediterranean University (EMU), Cyprus. His research interest is in Moving Target Defense (MTD), network security modeling and analysis, and cloud computing security, and cryptography.

**Dong Seong Kim** is an Associate Professor in School of Information Technology and Electrical Engineering, The University of Queensland (UQ), Brisbane, Australia. Prior to UQ, he led the Cybersecurity Lab at the University of Canterbury (UC), Christchurch, New Zealand from August 2011 to Jan 2019. He was a Senior Lecturer in Cybersecurity in the Department of Computer Science and Software Engineering at UC. He was a visiting scholar at the University of Maryland, College Park, Maryland in the US in 2007. From June 2008 to July 2011, he was a postdoc at Duke University, Durham, North Carolina in the US. His research interests are in cyber security and dependability for various systems and networks.

**Julian Jang-Jaccard** is an Associate Professor in School of Natural and Computational Sciences, Massey University, Auckland, New Zealand. Julian received her PhD in database transaction (University of Sydney); a Master of Information Engineering (University of Sydney) and a Bachelor of Computer Science (University of Western Sydney). Her research interests spans from database, cloud computing, mobile systems, cybersecurity, and big data analytics with a specific focus on security and privacy. She has many years of industrial strength application development experience and has been trained for an entrepreneurship skill. She has published numerous peerreviewed papers in respectable conferences and journals across a wide range of computing science and engineering communities.

**Declaration of interests**

☒ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: