

Article

Secure Cyber Deception Architecture and Decoy Injection to Mitigate the Insider Threat

Kyungmin Park ¹ , Samuel Woo ¹, Daesung Moon ¹ and Hoon Choi ^{2,*}

¹ Electronics and Telecommunications Research Institute, 218 Gajeong-ro, Yuseong-gu, Daejeon 34129, Korea; kmpark@etri.re.kr (K.P.); samuelwoo@etri.re.kr (S.W.); daesung@etri.re.kr (D.M.)

² Department of Computer Science & Engineering, Chungnam National University, 99 Daehak-ro, Yuseong-gu, Daejeon 34134, Korea

* Correspondence: hc@cnu.ac.kr; Tel.: +82-10-3410-6652

Received: 18 September 2017; Accepted: 29 November 2017; Published: 2 January 2018

Abstract: We propose a novel dynamic host mutation (DHM) architecture based on moving target defense (MTD) that can actively cope with cyberattacks. The goal of the DHM is to break the cyber kill chain, expand the attack surface to increase the attacker's target analysis cost, and disrupt the attacker's fingerprinting to disable the server trace. We define the participating entities that share the MTD policy within the enterprise network or the critical infrastructure, and define functional modules of each entity for DHM enforcement. The threat model of this study is an insider threat of a type not considered in previous studies. We define an attack model considering an insider threat and propose a decoy injection mechanism to confuse the attacker. In addition, we analyze the security of the proposed structure and mechanism based on the security requirements and propose a trade-off considering security and availability.

Keywords: moving target defense; network security; proactive security; decoy injection

1. Introduction

Moving target defense (MTD) is a type of security technology involving the novel concept of the IT infrastructure changing its form actively to prevent various types of cyberattacks. MTD is the most notable area of study in the White House publication entitled "Trustworthy Cyberspace: The Federal Cybersecurity Research and Development Program", published in 2011 [1]. Network address mutation is one of the most active research areas in MTD studies [2]. The network address mutation approach is a proactive defense strategy that disables cyberattacks by changing the main attributes of a network host (e.g., the IP address, port, network topology, platform, software). Before the concept of MTD was established, techniques that periodically change the host's network address related to mutations of network addresses were regularly investigated [3,4]. However, changing only the network address of the target host is not efficient because such changes can easily be tracked by the host's fingerprint when the active network address space is small [5]. Recently, MTD studies have proposed the concepts of fingerprint mutation or decoy node operation to address these limitations. The dynamic host mutation (DHM) architecture proposed in this paper consists of network address mutation for breaking the cyber kill chain, decoy node operation for increasing the attack cost by expanding the attack surface, and fingerprint mutation for disabling the server trace through the attacker's fingerprinting. We define the entities participating in the MTD policy to operate DHM and their relationships, and define the DHM functional module of each entity.

MTD technology, which encompasses network address mutation, fingerprint mutation and decoy node operation, can make target scanning and analysis much more difficult for an attacker than an existing static network security technology. However, existing MTD technologies are vulnerable to

the following two threats. The first threat occurs when an attacker infects a legitimate client that is constantly connected to a server protected by MTD technology. The second threat occurs when an attacker penetrates inside a subnet protected by MTD technology. Previous studies adequately suppressed threats from external attackers but did not consider the insider threats that arise in the two situations described above. An insider threat is caused by a drive-by-download attack due to user carelessness, or an attack that is executed in stages after determining the attack path using an attack graph [6–8]. In this paper, we define an attack model considering an insider threat and propose a deception mechanism to mitigate it.

The paper proceeds as follows. Related work is introduced in Section 2. The attack model considering an insider threat is defined in Section 3. The DHM architecture fused with the three MTD technologies, the security requirements for suppressing insider threats, and the decoy injection mechanism are introduced in Section 4. A security analysis of the proposed architecture and mechanism is discussed in Section 5, experimental results for the decoy injection mechanism are presented in Section 6, and conclusions are given in Section 7.

2. Related Work

The trends in MTD research in recent years are shown in Table 1. Typical examples of MTD research are openflow–random host mutation (OF–RHM) and random host mutation (RHM), which undertake network address mutation operations using SDN(Software Defined Networking) and legacy network operations with the use of virtual IPs [9,10]. HIDE (Host IDentify anonymization) is an MTD technology that uses a honeypot cloud based on RHM and adds the concepts of fingerprint mutation and decoy node operation [11]. Moving target IPv6 defense (MT6D) is a technology for generating addresses using a cryptographic algorithm with timestamps in an IPv6-based network environment [12]. Decoy-enhanced seamless IP randomization (DESIR) undertakes seamless connection migration to prevent service disconnections due to server address changes [13]. SDN-based fingerprint hopping is a technique by which to generate a false fingerprint during network domain scanning by an attacker [14].

Table 1. Trends in MTD research.

Research	Network Address Mutation	Fingerprint Mutation	Decoy Node Operation
OF–RHM & RHM	○		
HIDE	○	○	○
MT6D	○		
DESIR	○		○
SDN-Based Fingerprint Hopping	○	○	

2.1. OF–RHM & RHM

OF–RHM is a representative form of network address mutation that allocates virtual IPs to hosts and periodically translates them for protection against external attackers. OF–RHM maps the real IP and virtual IP of the host and reveals only the virtual IP while hiding the real IP from outside through packet-level processing. Owing to the periodic changing of the virtual IP, the attacker can never be certain about the information obtained through network scanning. A legitimate user can find the address of the server through DNS. Because a name service is used, the user can receive transparent service. The creation of the virtual IP and the determination of the address translation cycle are determined through a mutation controller. Given that OF–RHM is only applicable to SDN and lacks scalability, the OF–RHM research team has studied RHM applicable to legacy networks. The difference between OF–RHM and RHM is that OF–RHM performs IP translations using the OF-Switch and OF-Router SDN equipment, while RHM performs IP translation using a mutation gateway at the front of each subnet.

2.2. HIDE

The OF-RHM/RHM research team has been conducting ongoing research based on RHM [15–17]. HIDE, the latest research area of the RHM research team, includes fingerprint mutation and attack surface expansion, operating a honeypot cloud as well as network address mutation on the RHM network model. The honeypot cloud is located on a different network from the protected host. The gateway connected to the external network judges suspicious traffic. Any traffic arriving at addresses that are not active can be seen as suspicious traffic. This traffic is sent to the honeypot. This type of fingerprint mutation using a honeypot may not only present an attacker with some uncertainty about their scanning results but may also cause difficulty when an attacker attempts to analyze the degree of vulnerability and create attack weapons.

2.3. MT6D

MT6D (moving target IPv6 defense) is a network address mutation scheme that periodically changes the IID (Interface IDentifier) in a network using IPv6. In order to make it difficult for an attacker to know the address of the host when attacking, a one-way hash function is proposed in which a new address is generated by inputting the current IID and timestamp. MT6D, like RHM or HIDE, changes the address of hosts through a gateway. The difference between MT6D and the concepts in previous studies is that communication between gateways uses the UDP protocol rather than TCP to minimize the side effect of connection loss.

2.4. DESIR

DESIR is a technique that randomly translates the IP from an end-point host, unlike previous studies that changed the IP through SDN boxes or network gateways. DESIR operates decoy nodes to enhance the effect of IP translation in end-point hosts. Decoy-based MTD [18] is a technology that creates multiple virtual decoy nodes on the same network as protected servers, and continuously changes the addresses of real servers and decoy servers over time. The DESIR team applied the decoy-based MTD technology to attack scenarios to invalidate the blacklist of decoy nodes created during the target exploration phase. In order to mitigate the side effect of disconnections caused by changes of the address of the end-point server, a type of seamless connection migration technology is proposed in this study to ensure continuous service to legitimate users.

2.5. SDN Fingerprint Hopping

Due to the development of secure network boxes, studies of fingerprint randomization and concealment based on SDN using an intrusion detection system (IDS) and a fingerprint hopping engine have been conducted. These studies distinguish legitimate user traffic and suspicious traffic from an attacker through IDS and modify the headers of packets through a fingerprint-hopping engine for suspicious traffic as an OS other than the real server's OS. However, these studies have disadvantages in that they cannot adequately deal with fingerprinting by skilled attackers who cannot distinguish them through IDS. In addition to this fingerprint-hopping technology, there are also fingerprint-randomization techniques that manipulate ICMP (Internet Control Message Protocol) messages directly, but fingerprint randomization can cause unintended network service failures. In order to realize high-quality MTD technology, research on fingerprint mutation that can avoid network failures even in the absence of IDS is needed.

An open port on a server where a well-known port is always open to provide a specific service, such as a DB (DataBase) server or a Web server, can be used as a fingerprint by an attacker. Therefore, port-hopping techniques have been studied in an effort to make it difficult for an attacker to recognize a target through an open port [19–21].

3. Threat Model

A legitimate client using the MTD approach can continuously find a moving server. If a legitimate client is infected by an attacker, the attacker can easily find a moving server by analyzing the client's traffic or by checking the network state. Hence, previous MTD studies excluded the insider (=inside attacker) from the attack model. In this paper, an attack model is defined for a situation in which an attacker infects a legitimate client and where an attacker infiltrates a network on which a moving server exists. In this section, the protected domain by DHM and the attack methods enforceable by insiders are defined.

3.1. Protected Domain

In an enterprise network or ICT infrastructure, high-priority servers are placed on the internal network and protected. In this case, the threat of exposure to the server due to external scanning is insignificant. However, if a client on the internal network is infected by an attacker by means of a drive-by-download attack, the attacker will be able to access moving servers on the internal network. Figure 1 shows the protected domain, which is divided into four areas here.

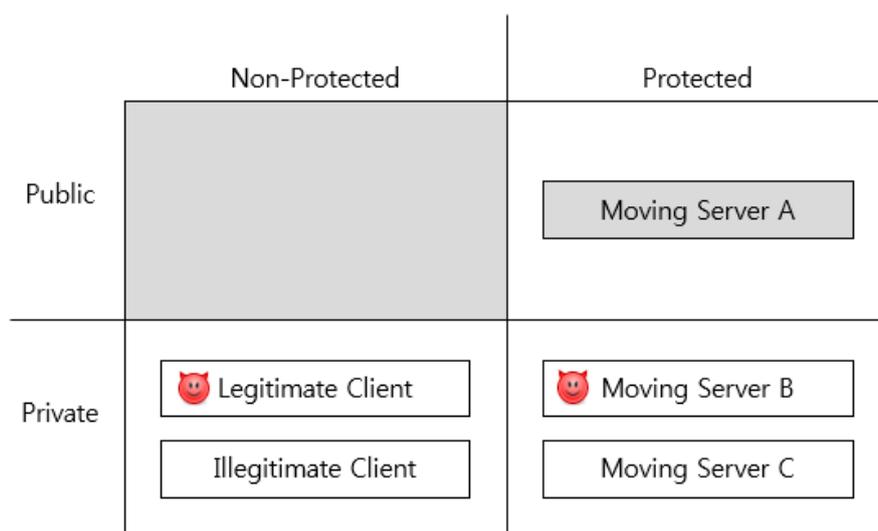


Figure 1. Protected domain.

The moving server A in the protected/public area is directly accessible from the outside. MTD technology to protect these types of servers is not mentioned in this paper because it has been studied extensively. The legitimate client in the non-protected/private area uses the MTD approach so as to access the moving servers in the protected/private area. The illegitimate client does not participate in MTD. Both of these clients can be infected by an attacker through a drive-by-download attack. When the illegitimate client is infected by an attacker, an attacker can detect the moving servers through internal network scanning. This case is not covered in this paper because it is identical to the case of an external attacker scanning to detect moving server A.

The scope of our threat model covered in this paper is a threat to the moving server B, which occurs when the legitimate client is infected, and a threat to the moving server C, which occurs when the moving server B is infected. Henceforth, these two threats are referred to as insider threats. It is assumed that communication between the moving server and the legitimate client is encrypted.

3.2. Attack Model

When the legitimate client is infected, an insider can easily detect the moving server in two ways. The first is by a traffic analysis. The legitimate client will have many network connections as

well as connections to the moving server. However, if the insider analyzes the inbound/outbound traffic of the legitimate client and finds a certain pattern, the insider will be able to keep track of the moving server. The second way is to run a utility (e.g., “netstat”) that can check the network status to verify established connections. The insider, who checks the network status, will analyze all of the connections to find the connection to the moving server. If the insider is lucky, the insider will be able to achieve this goal simply by checking the service name or a well-known port number bound to the connections without having to analyze the connections thoroughly. Moreover, an insider who knows that an attacked domain is protected by MTD can guess the address of the moving server by observing the external address of a specific process ID (PID) that changes periodically. In order to address these threats caused by certain privileged insiders, novel MTD mechanisms are required.

When the moving server B is infected, the insider has penetrated the subnet on which the final attack target exists. As in research on the RHM, MTD technologies that operate gateways on a per-subnet basis do not defend against insider attacks that penetrate the subnet where a moving server exists, because the gateways manipulate only the traffic passing through them and make the target appear to move. Similar to DESIR, MTD technology using address hopping in the end hosts cannot prevent a network traffic analysis by an insider using ARP (Address Resolution Protocol) spoofing.

In this paper, we propose decoy traffic injection and connection obfuscation mechanisms to cope with these two attack models.

4. Design

In Section 4.1, security requirements based on the attack model defined in Section 3.2 are defined. In Section 4.2, relationships between entities participating in the DHM approach, software architecture, and detailed functional modules are proposed. Finally, in Section 4.3, mechanisms to mitigate insider threats are proposed.

4.1. Security Requirement

In Section 3, an attack model considering an insider threat, which was excluded from previous studies, is defined. Because MTD requirements for external attackers were proposed in previous studies, we define only the security requirements for an insider threat, as follows.

Requirement 1. The decoy traffic-injection mechanism should make it as difficult as possible to analyze the traffic to find the moving server by the insider who infected the legitimate client.

Requirement 2. The decoy traffic-injection mechanism makes it difficult to analyze the traffic to find the moving server by an insider who penetrated the subnet on which the moving server exists.

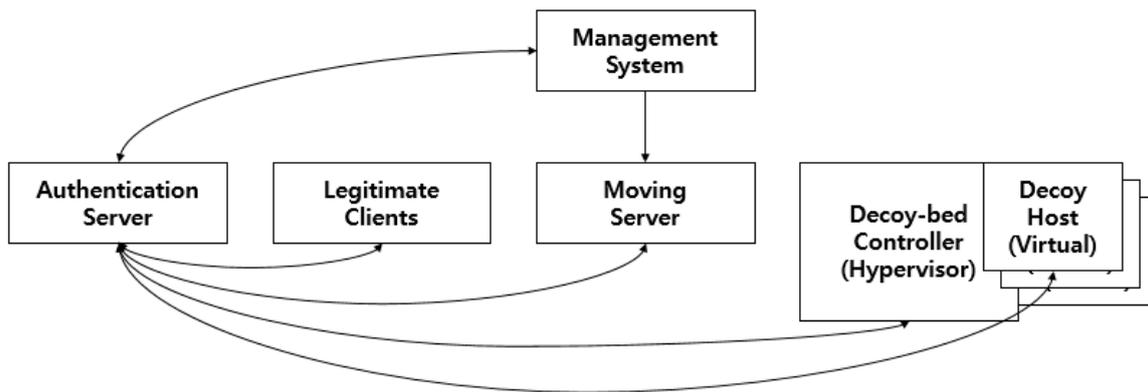
Requirement 3. The connection obfuscation mechanism makes it extremely difficult for an insider who infected the legitimate client to find the moving server through the network status and by active process monitoring.

Requirement 4. The DHM should prevent the continuous tracking of the moving server through MAC (Media Access Control) address analysis and fingerprinting by the insider who penetrated the subnet on which the moving server exists.

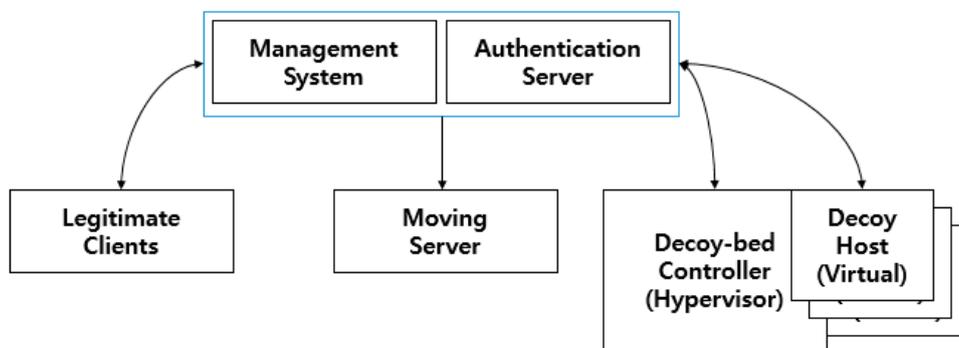
The defined security requirements are intended to mitigate the threats that arise within the network infrastructure in which MTD is being implemented. The stationary security system uses security devices such as an IDS or attack detection mechanisms in application layer [22,23] to detect an attacker who has penetrated the inside. Unlike stationary security systems, MTD aims to deceive the attacker through constant network mutations rather than detecting an insider. Naturally, the best security policy is an instance of combined use with the existing IDS with practical MTD technology, but we focus solely on MTD technology to deceive attackers here.

4.2. Architecture

The DHM network entities and their relationships proposed in this paper are shown in Figure 2. The management system collects information about the moving server and the network to which it belongs, performs the attack point analysis, and then determines the mutation strategy. The mutation strategy consists of network address mutation, fingerprint mutation, and decoy node operations. The decoy host expands the attack surface to increase the cost incurred by an attacker to recognize a moving server. The decoy-bed controller is responsible for creating and operating decoy hosts according to the decoy operation received from the management system. Because the moving server and the decoy host share the address space within the same network, they must share the mutation rule to avoid address collisions during the mutation process.



(A) Separate management server and authentication server



(B) Integrated management server and authentication server

Figure 2. DHM entities for MTD.

The legitimate client using the services provided by the moving server must be aware of the mutation rule so that it can connect to the moving server. Mutated addresses are generated based on a secret key. Because the secret key must be distributed only to the entities sharing the mutation rule, the authentication server authenticates the entities and distributes the secret keys. After authentication, the entities that have acquired the secret key perform address mutation using a crypto function. A counter or timestamp can be used to synchronize addresses between entities.

A client with a secret key must first be authenticated through an authentication server to access the moving server. Similar to DNS, the authentication server sends the address of the moving server to the authenticated legitimate client. If the authentication method is not robust, the attacker will intercept the MTD policy. In this regard, recently, Bayesian classifiers using question/response patterns have been studied to authenticate legitimate users [24]. Also, since the authentication server performs an operation similar to the DNS, techniques such as the latest research that provide DNS security based on a public key infrastructure rather than a certificate basis [25] can be applied in the authentication server.

The management system is functionally separated from the authentication server, as shown in Figure 2A, but it can operate with the authentication server in a physical machine, as shown in Figure 2B.

The moving server, decoy host, and legitimate client must be equipped with a software module for DHM. In this paper, we define the software module as a self-mutation system. The subsystem configuration of the self-mutation system of each entity is shown in Figure 3.

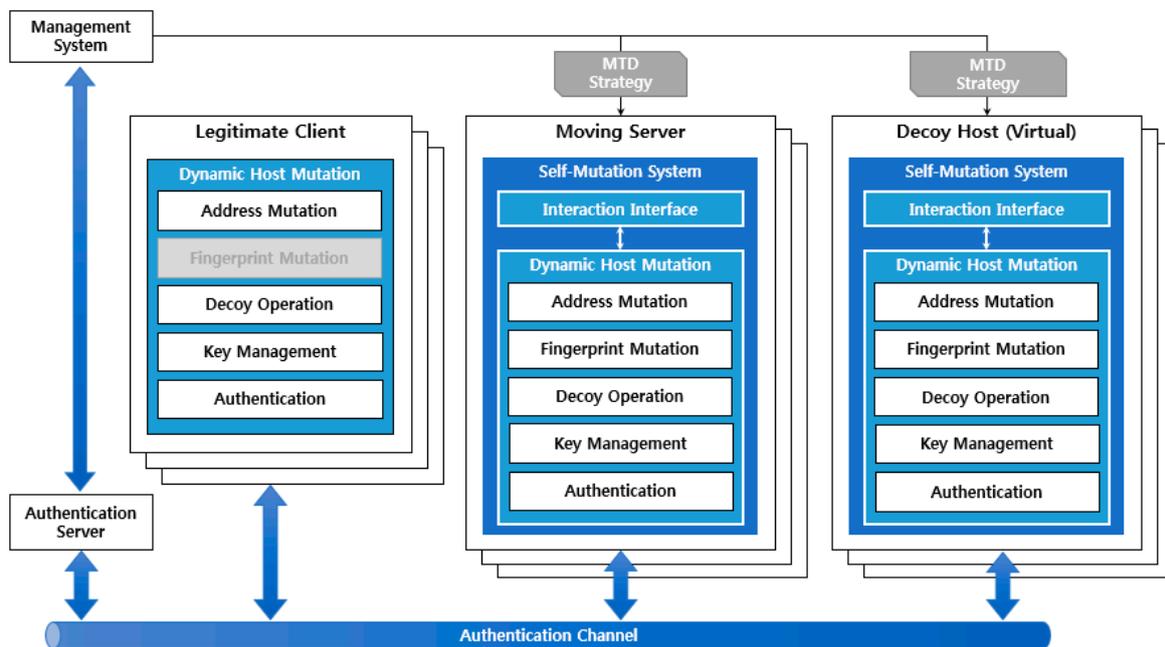


Figure 3. DHM entities for MTD.

The authentication and key management are responsible for authentication and key distribution for all entities [24,25]. The moving server and decoy host have interfaces for interactions by which to receive the mutation strategy from the management system. The decoy host needs the address mutation because it has to undergo mutation, like the moving server. The legitimate client does not mutate, but it needs to know the changed address of the moving server. Hence, it only knows the address generation rule through the address mutation.

Because a skilled attacker can identify a target through a fingerprint trace, the moving server must undergo fingerprint mutation. The decoy host needs the fingerprint mutation because it can be excluded from the analysis if the decoy host is traced by fingerprinting.

Table 2 is a description of the function modules constituting the five subsystems of the self-mutation system.

Table 2. Descriptions of function modules.

Subsystem	Module	Description
Network Address Mutation	Crypto Function-Based Address Generation	Address generation function using a cryptographic algorithm for address synchronization without interactions between entities. The algorithm that generates the same output using the same input and correct output values only for entities sharing symmetric keys.
	IP Collision Detection	Conflict detection for addresses generated by each host.
	Connection Migration	Connection migration to prevent a service disruption while a host address is being changed.
	Network Address Shuffling	Efficient MAC learning and updating ARP table while a host address is being changed.
Fingerprint Mutation	Activity Monitor	Monitoring the status of connections between a server and clients.
	OS Fingerprint Mutation	Translating various fingerprints collected through network scanning by an attacker.
	Fault Management	Forecasting any faults due to a fingerprint mutation.
Decoy Operation	Node Generation	Creating virtual nodes to increase the active address space to prevent recognition of the target by an attacker.
	Traffic Generation	Creating decoy traffic to prevent recognition of the target when an attacker who penetrates the internal network analyzes network traffic.
	Connection Generator	Creating decoy connections to prevent recognition of the target by analyzing network status by an attacker who penetrates the internal network.
	Context Awareness	Collecting attributes to create decoy nodes similar to moving servers.
Host Authentication	Certificate Based Auth.	Entity authentication based on certificates.
	Symmetric Key Based Auth.	Entity authentication based on a symmetric key.
Key Management	Key Distribution	Session key distribution used for address generation.
	Key Update	Ensuring session key safety.
	Key Revocation	

4.3. Decoy Injection

A typical study that was conducted for the purpose of mitigating an insider threat is that which developed the indistinguishable decoy injection method [26,27]. In this line of research, the proposed method automatically generates indistinguishable decoy traffic in the wireless network to prevent an attacker who accesses the wireless network from stealing and abusing network traffic. The research is somewhat different from our research domain because it was conducted on a subnet of a wireless network, where various terminal nodes exist, but we were motivated by the study to devise mechanisms to mitigate an insider threat.

The two types of insiders mentioned in Section 3 analyze traffic and the network connection status to detect the moving server. A key idea of MTD is to change the information the attacker analyzes to identify the target continuously. As a result, the MTD technology which responds to an insider threat must deceive the insider by changing the traffic and network connection status. The connection obfuscation and the traffic injection steps described below are both part of the decoy node operation of the DHM architecture because they involve interaction with decoy servers.

4.3.1. Connection Obfuscation

An insider who has infected the legitimate client can detect the moving server simply by executing a simple command. Taking an extreme situation as an example, if the legitimate client has only a

connection to the moving server, the insider will have no difficulty in verifying that the moving server is the target the insider is attempting to attack through a vulnerability analysis.

The first step in the connection obfuscation is, as shown in Figure 4, deliberately to increase the number of legitimate client connections that the insider must analyze to detect the target. Henceforth in this paper, these connections are called decoy connections. Decoy connections are between the legitimate client and the decoy server. In previous studies that did not consider insider threats, the decoy server was used only to expand the attack surface, whereas in this study, which considers insider threats, decoy servers should establish decoy connections with the legitimate clients.

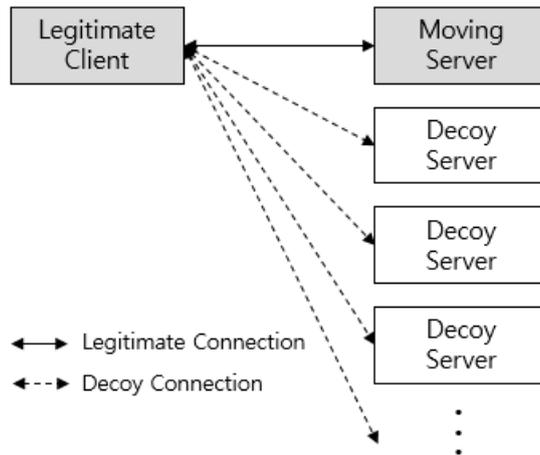


Figure 4. An example of decoy connections.

For MTD administrators, it is possible to determine which connections are decoy connections, but the insider will look at all connections during their analysis. Simply by establishing decoy connections in the legitimate client, the insiders’ analysis cost can be increased, as shown in Figure 5.

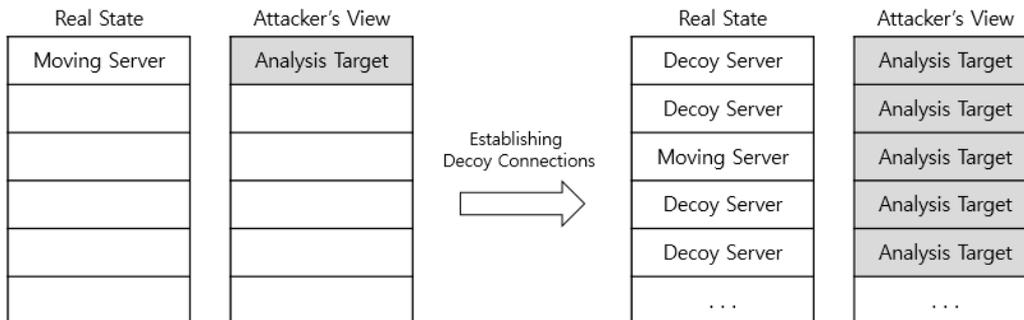


Figure 5. Analysis cost increase due to decoy connections.

There are several drawbacks to this approach. If the network state does not mutate when decoy connections are established, the insider’s attack will not be different from an attack on the stationary network infrastructure. Also in this state, if only the external address of the legitimate connection changes continuously due to the network address mutation by the moving server, the address mutation technology of the MTD informs the attacker of the moving server. These drawbacks are solved by address shuffling of the moving server and by using decoy servers, as in previous dynamic decoy studies [28,29]. Nevertheless, there are still fatal drawbacks when using this method.

An insider can associate network connections with the process information (e.g., PID, process name, memory consumption) bound to it through simple command executions. Most MTD technologies guarantee transparency of the application layer even if the network configuration is

mutated. Therefore, even if the moving server address mutates, as shown in Figure 6, the process of establishing the connection does not change. A fixed PID can be used as a fingerprint by an insider. The insider will be able to analyze the static factor rather than analyzing the dynamic factor.

If an insider analyzes the processes using PID and strace (the system call debugging utility in Linux) instead of analyzing the dynamic factor, the insider may associate the PID with a legitimate process or a decoy process. If the PID of the legitimate process is exposed to the insider by means of a process analysis, the insider will be able to confirm the external address related to the PID as the moving server. Moreover, if the insider who analyzes the dynamic factor detects the moving server even once in a certain time period, from the next period the insider will be assured that the external address of the process that was associated with the moving server during the previous period is the address of the moving server.

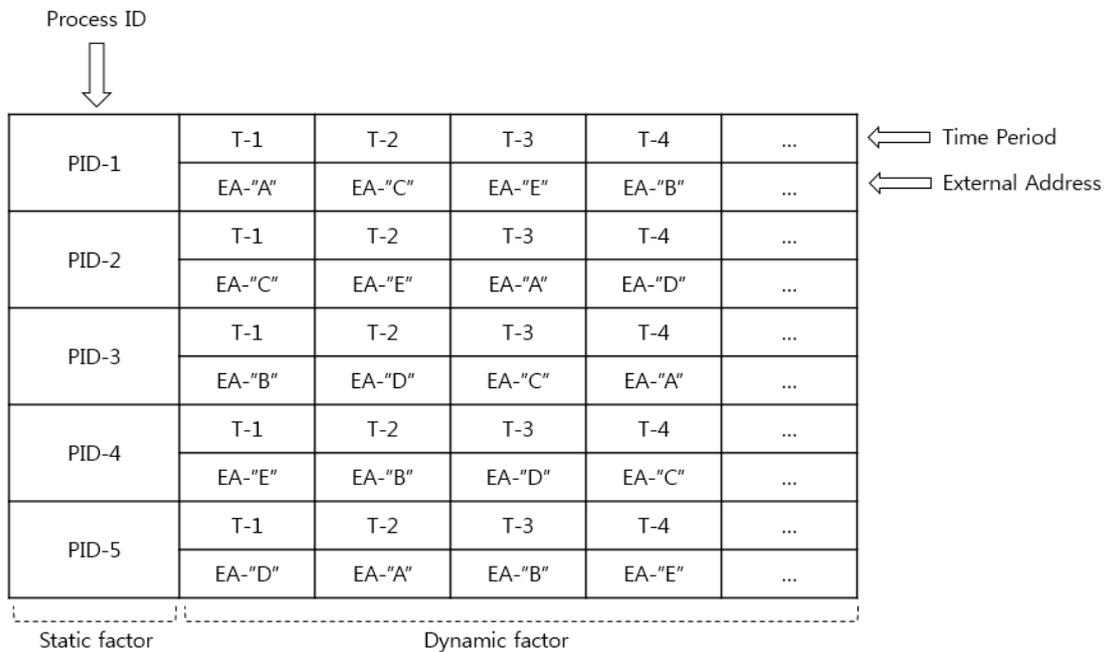


Figure 6. The dynamic address mutation & static PID.

This drawback can be solved by PID randomization. As an example of the Windows operating system, EPROCESS, an object for managing process information in the kernel, can be used. The EPROCESS structure stores the memory location of the PID of a specific process. If the value of this location changes, the PID of the running process also changes. As a result, if PID shuffling is enforced simultaneously with address shuffling, the insider attack described above is mitigated. However, there is a need for more in-depth study of the side effects and overhead analysis caused by the random shuffling of the PIDs of running processes.

Other information that an insider can use to detect the moving server is the memory usage of the process bound to the connection. If the insider continues to analyze this memory usage, because the memory usage can be used to detect the moving server as well as the PID, a context-awareness decoy operation is required to control the memory usage of the decoy process properly.

4.3.2. Decoy Traffic Injection

The connection obfuscation approach described above is a mechanism for mitigating the threat from an infected legitimate client. However, if an attacker analyzes network traffic, it is useless to replicate only the connection. Hence, a decoy traffic injection is required. The decoy traffic injection can mitigate not only threats from insiders who have infected the legitimate client, but also threats

from insiders who have penetrated the same subnet as the moving server, as mentioned in Chapter 3. Because the moving server and the legitimate clients use encrypted communication, insiders cannot analyze the contents of payloads, but they can find the moving server by analyzing the traffic flow and frequency of occurrence. The first goal of a decoy traffic injection is to increase the amount of traffic an attacker must analyze. In order to achieve this goal alone, it is sufficient to generate decoy traffic irregularly. In this case, however, a sophisticated insider can easily distinguish between decoy traffic and legitimate traffic, which can greatly reduce the cost of the analysis. The second goal of a decoy traffic injection is to mimic the legitimate traffic pattern to create more efficient decoy traffic. To achieve the second goal, context awareness by the decoy traffic injection as proposed in this paper is shown in Figure 7.

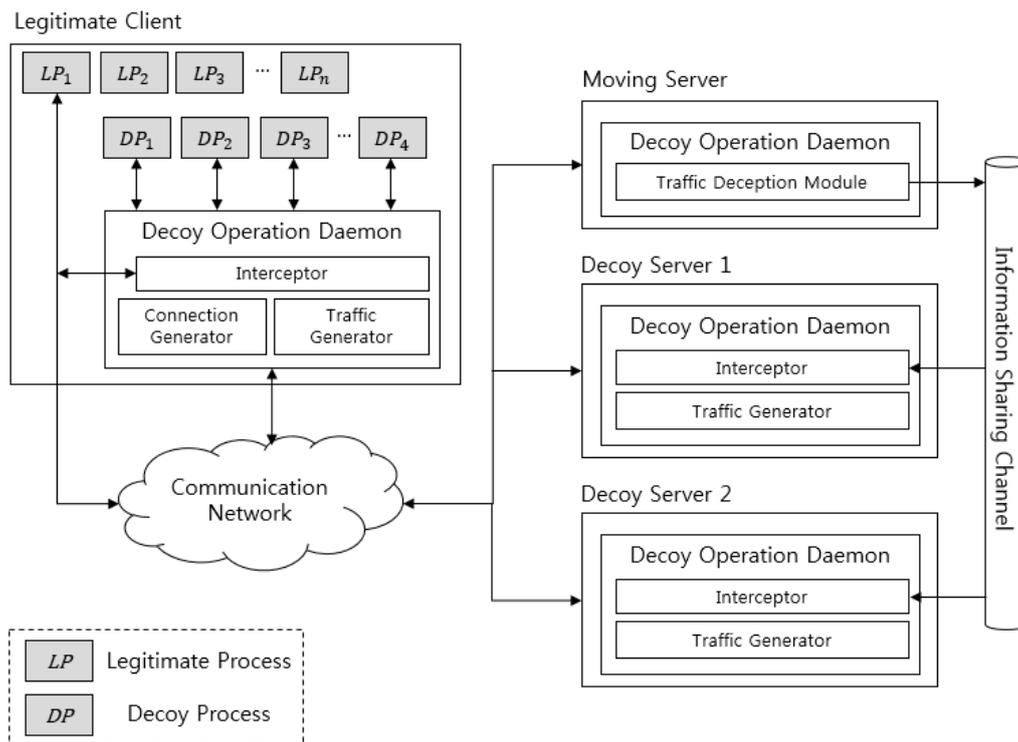


Figure 7. A mechanism of decoy traffic injection with context awareness.

The LP (legitimate process) is the process bound to the legitimate connection, and the DP (decoy process) is the process bound to the decoy connection. The decoy connection is created by the connection-generator module of the decoy operation daemon. The connection generator module refers to the LP behavior through the interceptor module to create decoy connections. The interceptor module of the decoy operation daemon monitors the outbound traffic generated by the LP. The traffic generator module requires that traffic generated to the DP is created as LP. The generated traffic shows a pattern similar to that of the outbound traffic generated by the LP, but the two are not identical. The decoy traffic generated by the DP is not always generated after the LP generates traffic. The traffic generator analyzes the pattern of traffic generated by the LP and determines the traffic generation policy within a range determined to be similar to it.

The traffic generator module of the decoy server mimics the outbound traffic of the moving server and generates traffic between the decoy server and the DP. The traffic deception module of the moving server shares the characteristics of the outbound traffic generated by the moving server to decoy servers through a secure channel. The traffic generator module of the decoy server, like the traffic generator module of the legitimate client, analyzes the pattern of traffic generated by the moving server and determines a policy within a range that is judged to be similar.

4.3.3. Limitation

The decoy injection mechanism is a countermeasure to mitigate the insider threat. However, if a decoy connection and traffic occur frequently, degradation of the system and network performance is inevitable. Therefore, it is inefficient to use the decoy injection strategy in network infrastructures where large amounts of traffic are generated. Because the number of legitimate clients also affects performance, it is desirable to apply the decoy injection only after considering trade-offs related to the security and availability of the network infrastructure. This paper does not mention IDS, which is a traditional security system, but the decoy injection can be applied to critical infrastructures where both security and availability should be considered, even if a low degree of availability is suspected only when an insider threat is suspected from IDS.

5. Security Analysis

In this section, we analyze the security of the proposed MTD structure and the decoy injection method. A security analysis is performed based on the security requirements defined in Section 3.

- Reqs. 1 and 2: In order to increase the cost of the traffic analysis necessary to detect the moving server, the proposed traffic-injection mechanism increases the amount of traffic that an attacker must analyze. In addition, because the decoy traffic is imitated by legitimate traffic, the analysis results of the insider are uncertain.
- Req. 3: In Section 4.3.1, we analyzed various types of network state information and process information that an insider can use to detect the moving server. Based on the analysis results, we propose the connection-obfuscation mechanism based on the MTD concept of continuously changing the information used by the insider.

In the situation where one legitimate connection exists and the connection obfuscation for this is being applied, the initial probability (P_i) that the insider will recognize the moving server is

$$P_i = \frac{1}{1 + \text{number of decoy connections}}.$$

Because the sophisticated insiders can precisely analyze the decoy connections and exclude them from an attack list, the probability that an insider will recognize a moving server at a particular time t (P_t) is

$$P_t = \frac{1}{1 + \text{number of decoy connections} - \text{number of excluded connections}}$$

(number of excluded connections \leq number of decoy connections).

Over time, since P_t goes to 1, if the insider can get enough time, using only the connection obfuscation will not protect the moving server. However, if the addresses of the moving server and the decoy servers are periodically shuffled, P_t will be initialized to P_i , so the insider will have to resume trying to identify the moving server. The decoy traffic injection makes it difficult for the insider to judge the excluded decoy connection.

In summary, the decoy injection mechanism proposed in this paper inhibits the increase of P_t through the decoy traffic injection, and periodically initializes P_t to P_i through the connection obfuscation. By doing so, the decoy injection mechanism meets the security requirements 1–3, which are related to increased attack cost.

- Req. 4: MTD technology to operate the gateway in front of the network switch is fatal to the threat of an attacker who has penetrated the subnet on which the moving server is located. Because the proposed DHM architecture is implemented in an end-point host, it is not easy for an insider to detect the moving server when it is protected by decoy servers and end-point hopping, even if there is an insider on the subnet. In addition, DHM can further increase the complexity of an

attack by adding MAC address and OS fingerprinting information (e.g., TTL values, window sizes) to end-point hopping rules through mutual authentication between the entities participating in MTD.

6. Experiment

In order to verify the feasibility of the decoy injection mechanism, we implemented a client-side decoy operation daemon and measured the consistency between the legitimate traffic and the decoy traffic. The decoy traffic should not be exactly the same as the legitimate traffic, nor be too different. As described in Section 4.3.2, in order not to be easily distinguished by a skilled insider, the decoy traffic should look similar to the legitimate traffic and have an irregular pattern.

The interceptor module of the decoy operation daemon implemented in this experiment calculates the total amount of legitimate traffic generated over a period of time. The traffic generator generates the decoy traffic for the next time period based on the total amount of legitimate traffic calculated in the previous time period. The decoy traffic generated by the traffic generator has another pattern similar to the legitimate traffic pattern. In this experiment, we applied the method of randomizing the decoy traffic during the next time period by the total amount of legitimate traffic calculated over a previous time period. The experimental environment and scenarios are shown in Figure 8.

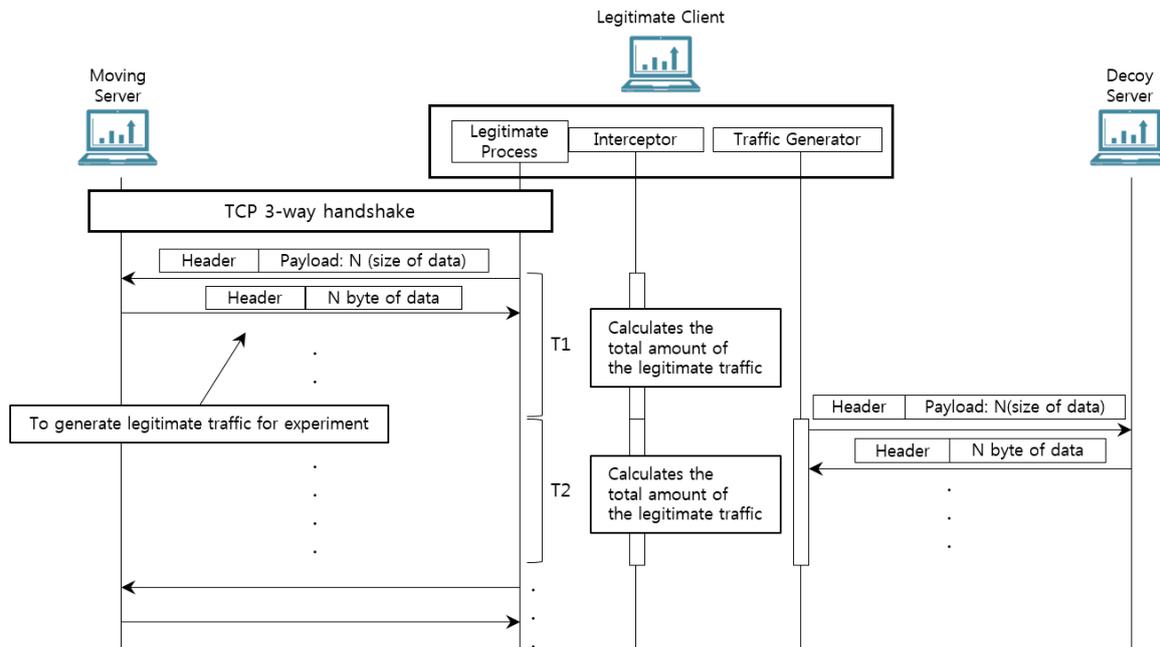


Figure 8. The experimental environment and scenarios.

In order to generate the legitimate traffic for the experiment, the legitimate client sends an integer value N to the moving server every 3 s, and the moving server that receives this packet creates an N -byte-sized packet (payload size only) and sends it to the legitimate client. The integer value increases by 1 every 3 s within the range of 20–100, and decreases by 1 when it reaches the maximum value. In this experiment, the traffic generated as above is regarded as the legitimate traffic. The interceptor module monitors the ports that are communicating with the moving server through tcpdump and calculates the total amount of legitimate traffic. For intuitive consistency comparisons, traffic computed via tcpdump only considers traffic generated by experimental applications.

The purpose of this experiment is to examine the effect of the pattern-recording time period on legitimate traffic and decoy traffic consistency for legitimate traffic with a certain pattern. We also

expect another variable to be used as a new variant of deceiving insiders. Figure 9 shows decoy traffic for the ground truth with the time period of 6 s.

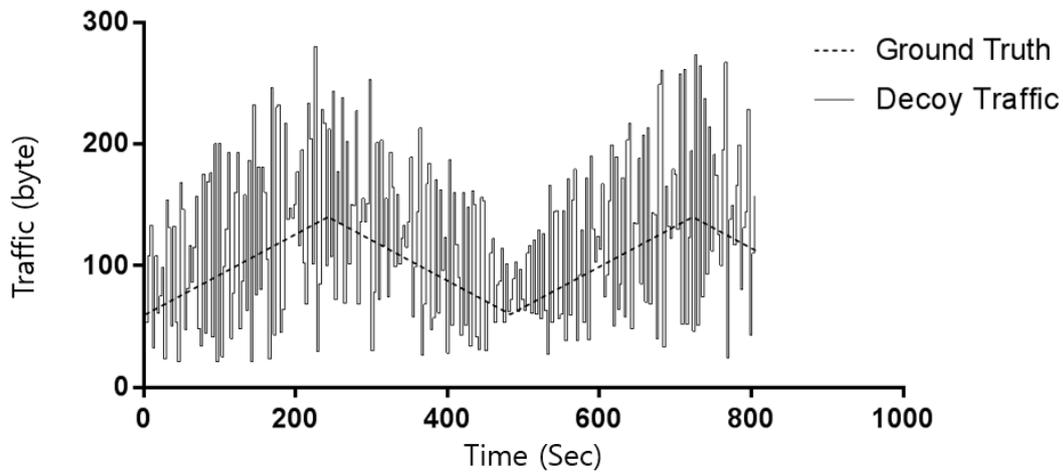


Figure 9. Comparison between the decoy traffic and the ground truth.

The ground truth is the phase of the legitimate traffic with a certain pattern over time. In the experiment in Figure 9, the traffic generator randomly generates the decoy traffic within the next time period by the amount of legitimate traffic generated in the previous time period. As you can see, although the decoy traffic seems to be very consistent with the ground truth, it is not identical. In order to increase the consistency between the decoy traffic and the ground truth, we applied the decoy traffic randomly but reduced the displacement. Figure 10 shows the decoy traffic with reduced displacement. In order to reduce the displacement of decoy traffic, we adjusted the range of randomly generated decoy traffic to approach the average value.

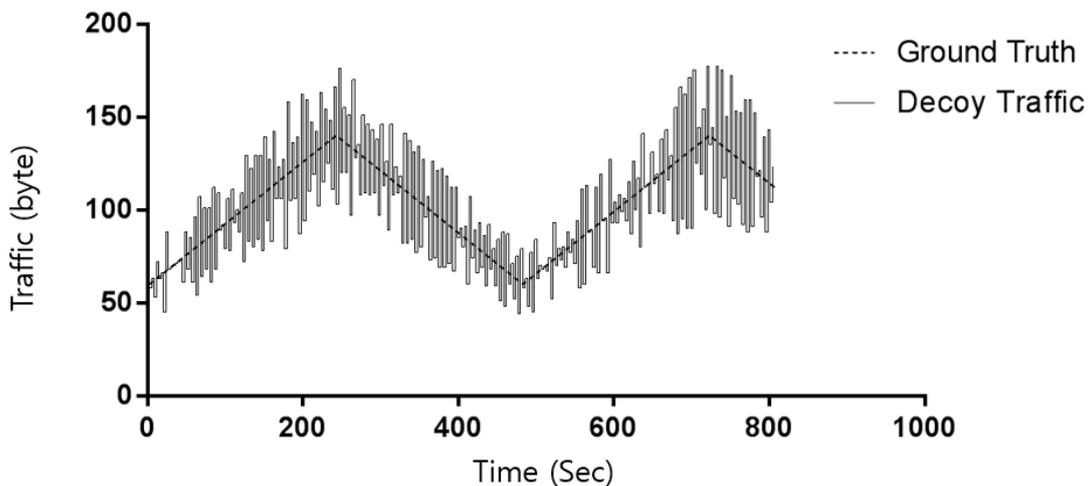


Figure 10. Consistency with reduced displacement.

As can be seen from the experimental results, when decoy traffic is generated, adding a slight deformation may increase or decrease consistency with the ground truth. If the pattern of decoy traffic does not change, it is desirable that the traffic generator periodically changes the shape of decoy traffic, because skilled insiders can easily distinguish decoy from APT (Advanced Persistent Threat) attacks.

7. Conclusions

In the traditional stationary network infrastructure, attackers who attack arbitrary network targets absolutely have advantages over defenders. Reactive and passive network security techniques cannot effectively defend against attacks in an environment of unforeseen vulnerabilities because these techniques allow attackers time to analyze targets. MTD is a technology that attracts attention in the field of cybersecurity because it can offer defenders advantages, unlike existing network security technologies. MTD should provide various network randomization techniques which can be used to deceive attackers. DHM moves the protected host continuously through network address mutations and disables attacks that track fingerprints through fingerprint mutation. The decoy node operation strategy aims to increase the target analysis cost of the attacker by expanding the attack surface. In this paper, we propose a DHM architecture that combines the three aforementioned MTD technologies. Because DHM enforces MTD with the end-point hopping method, it can cope with not only threats which come from the outside of the network, but also those posed by an insider penetrating inside. Specifically, the decoy injection mechanism implemented through the decoy node operation can mitigate insider threat that was not considered in previous studies. All MTD technologies generate various types and amounts of overhead. Therefore, in order to use a decoy injection effectively, the characteristics of the network domain considering the trade-off between security and network/system overhead must be considered as well.

Acknowledgments: This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No. 2017-0-00213, Development of Cyber Self Mutation Technologies for Proactive Cyber Defence).

Author Contributions: Kyungmin Park and Samuel Woo proposed the main structure and mechanism; Daesung Moon conceived and designed the experiments; Kyungmin Park and Hoon Choi performed the experiments and analyzed the data; Hoon Choi performed mathematical security analysis; Kyungmin Park wrote the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. NSTC. *Trustworthy Cyberspace: Strategic Plan for the Federal Cybersecurity R&D Program*; National Science and Technology Council: Washington, DC, USA, 2011.
2. Cai, G.; Wang, B.; Wang, X.; Yuan, Y.; Li, S. An introduction to network address shuffling. In Proceedings of the IEEE International Conference on Advanced Communication Technology (ICACT), Pyeongchang, Korea, 31 January–3 February 2016.
3. Antonatos, S.; Akritidis, P.; Markatos, E.P.; Anagnostakis, K.G. Defending against hitlist worms using network address space randomization. *Comput. Netw.* **2007**, *51*, 3471–3490. [[CrossRef](#)]
4. Kewley, D.; Fink, R.; Lowry, J.; Dean, M. Dynamic approaches to thwart adversary intelligence gathering. In Proceedings of the DARPA Information Survivability Conference & Exposition II, Anaheim, CA, USA, 12–14 June 2001.
5. Rahman, M.A.; Manshaei, M.H.; Al-Shaer, E. A game-theoretic approach for deceiving remote operating system fingerprinting. In Proceedings of the IEEE Communications and Network Security (CNS), National Harbor, MD, USA, 14–16 October 2013.
6. Kaynar, K.; Sivrikaya, F. Distributed attack graph generation. *IEEE Trans. Dependable Secur. Comput.* **2016**, *13*, 519–532. [[CrossRef](#)]
7. Cook, K.; Shaw, T.; Hawrylak, P.; Hale, J. Scalable Attack Graph Generation. In Proceedings of the 11th ACM Annual Cyber and Information Security Research Conference, Oak Ridge, TN, USA, 5–7 April 2016.
8. Johnson, P.; Verotte, A.; Ekstedt, M.; Lagerström, R. pwnpr3d: An attack-graph-driven probabilistic threat-modeling approach, Availability. In Proceedings of the IEEE International Conference on the Reliability and Security (ARES), Salzburg, Austria, 31 August–2 September 2016.
9. Jafarian, J.H.; Al-Shaer, E.; Duan, Q. Openflow random host mutation: Transparent moving target defense using software defined networking. In Proceedings of the ACM Workshop on Hot Topics in Software Defined Networks, Helsinki, Finland, 13 August 2012.

10. Al-Shaer, E.; Duan, Q.; Jafarian, J.H. Random Host Mutation for Moving Target Defense. In *SecureComm 2012: Security and Privacy in Communication Networks*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 310–327.
11. Jafarian, J.H.; Niakanlahiji, A.; Al-Shaer, E.; Duan, Q. Multi-Dimensional Host Identity Anonymization for Defeating Skilled Attackers. In *Proceedings of the ACM CCS Workshop on Moving Target Defense*, Vienna, Austria, 24 October 2016.
12. Dunlop, M.; Groat, S.; Urbanski, W.; Marchany, R.; Tront, J. Mt6d: A moving target ipv6 defense. In *Proceedings of the IEEE Military Communications Conference*, Baltimore, MD, USA, 7–10 November 2011.
13. Sun, J.; Sun, K. DESIR: Decoy-enhanced seamless IP randomization. In *Proceedings of the 35th Annual IEEE International Conference on Computer Communications (INFOCOM)*, San Francisco, CA, USA, 10–14 April 2016.
14. Zhao, Z.; Liu, F.; Gong, D. An SDN-Based Fingerprint Hopping Method to Prevent Fingerprinting Attacks. *Secur. Commun. Netw.* **2017**, *2017*. [[CrossRef](#)]
15. Jafarian, J.H.; Al-Shaer, E.; Duan, Q. Spatio-temporal address mutation for proactive cyber agility against sophisticated attackers. In *Proceedings of the ACM CCS Workshop on Moving Target Defense*, Scottsdale, AZ, USA, 7 November 2014.
16. Jafarian, J.H.; Al-Shaer, E.; Duan, Q. Adversary aware IP address randomization for proactive agility against sophisticated attackers. In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, Kowloon, Hong Kong, China, 26 April–1 May 2015.
17. Jafarian, J.H.; Al-Shaer, E.; Duan, Q. An effective address mutation approach for disrupting reconnaissance attacks. *IEEE Trans. Inf. Forensics Secur.* **2015**, *10*, 2562–2577. [[CrossRef](#)]
18. Clark, A.; Sun, K.; Poovendran, R. Effectiveness of IP address randomization in decoy-based moving target defense. In *Proceedings of the IEEE 52nd Annual Conference on Decision and Control (CDC)*, Florence, Italy, 10–13 December 2013.
19. Ma, D.; Lei, C.; Wang, L.; Zhang, H.; Xu, Z.; Li, M. A Self-adaptive Hopping Approach of Moving Target Defense to thwart Scanning Attacks. In *Information and Communications Security*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 39–53.
20. Shi, L.; Jia, C.; Lü, S.; Liu, Z. Port and address hopping for active cyber-defense. In *Intelligence and Security Informatics*; Springer: Berlin/Heidelberg, Germany, 2007; pp. 295–300.
21. Luo, Y.B.; Wang, B.S.; Wang, X.F.; Hu, X.F.; Cai, G.L.; Sun, H. RPAH: Random port and address hopping for thwarting internal and external adversaries. In *Proceedings of the IEEE Trust-com/BigDataSE/ISPA*, Helsinki, Finland, 20–22 August 2015.
22. Nagpal, B.; Chauhan, N.; Singh, N. A Survey on the Detection of SQL Injection Attacks and Their Countermeasures. *J. Inf. Process. Syst.* **2017**, *13*, 689–702.
23. Rathore, S.; Sharma, P.K.; Park, J.H. XSSClassifier: An Efficient XSS Attack Detection Approach Based on Machine Learning Classifier on SNSs. *J. Inf. Process. Syst.* **2017**, *13*, 1014–1028. [[CrossRef](#)]
24. Albayram, Y.; Khan, M.M.H.; Bamis, A.; Kentros, S.; Nguyen, N.; Jiang, R. Designing challenge questions for location-based authentication systems: A real-life study. *Hum. Centric Comput. Inf. Sci.* **2015**, *5*, 1–28. [[CrossRef](#)]
25. Im, H.; Kang, K.; Park, J.H. Certificateless based public key infrastructure using a DNSSEC. *J. Convergen.* **2015**, *6*, 26–33.
26. Bowen, B.M.; Kemerlis, V.P.; Prabhu, P.; Keromytis, A.D.; Stolfo, S.J. Automating the injection of believable decoys to detect snooping. In *Proceedings of the ACM Conference on Wireless Network Security*, Hoboken, NJ, USA, 22–24 March 2010.
27. Bowen, B.M.; Kemerlis, V.P.; Prabhu, P.; Keromytis, A.D.; Stolfo, S.J. A system for generating and injecting indistinguishable network decoys. *J. Comput. Secur.* **2012**, *20*, 199–221. [[CrossRef](#)]
28. Fan, W.; Fernández, D.; Du, Z. Versatile virtual honeynet management framework. *IET Inf. Secur.* **2016**, *11*, 38–45. [[CrossRef](#)]
29. Achleitner, S.; Porta, T.L.; McDaniel, P.; Sugrom, S.; Krishnamurthy, S.V.; Chadha, R. Cyber Deception: Virtual Networks to Defend Insider Reconnaissance. In *Proceedings of the ACM CCS International Workshop on Managing Insider Security Threats*, Vienna, Austria, 28 October 2016.

