Hindawi
Security and Communication Networks
Volume 2017, Article ID 1560594, 12 pages
https://doi.org/10.1155/2017/1560594

Research Article

An SDN-Based Fingerprint Hopping Method to Prevent Fingerprinting Attacks

Zheng Zhao, Fenlin Liu, and Daofu Gong

Zhengzhou Science and Technology Institute, Zhengzhou 450002, China

Correspondence should be addressed to Fenlin Liu; liufenlin@vip.sina.com

Received 3 November 2016; Revised 12 January 2017; Accepted 31 January 2017; Published 22 February 2017

Academic Editor: Roberto Di Pietro

Fingerprinting attacks are one of the most severe threats to the security of networks. Fingerprinting attack aims to obtain the operating system information of target hosts to make preparations for future attacks. In this paper, a fingerprint hopping method (FPH) is proposed based on software-defined networks to defend against fingerprinting attacks. FPH introduces the idea of moving target defense to show a hopping fingerprint toward the fingerprinting attackers. The interaction of the fingerprinting attack and its defense is modeled as a signal game, and the equilibriums of the game are analyzed to develop an optimal defense strategy. Experiments show that FPH can resist fingerprinting attacks effectively.

1. Introduction

Fingerprinting is a technique that is used to identify the operating system (OS) type and version of a target host and is an essential step for a successful network attack. With the OS information of the target host, the attacker can launch a better-targeted attack. Therefore, fingerprinting attacks are a significant threat to network security.

Fingerprinting attacks explore the OS of a target host based on the traffic from the target host. Different OS implementations and TCP/IP stacks exist; thus, different OS platforms communicate in different patterns, which means that some fields in packet headers are different and can be precisely distinguished by the fingerprinting attacker. The fingerprinting technique can be classified into two main classes: passive fingerprinting and active fingerprinting. A passive fingerprinting attacker sniffs and analyzes traffic from the target hosts and determines the OS type. Reconnaissance tools, such as p0f [1] and SinFP [2], can support this type of fingerprinting, whereas an active fingerprinting attacker sends a set of carefully constructed probes to the target host proactively and collects the response packets to determine the host OS type. Reconnaissance tools, such as Nmap [3] and Xprobe2 [4, 5], can be used in active fingerprinting. An attacker can collect much more OS information using active fingerprinting than passive fingerprinting, but active fingerprinting is more likely to be detected by a defender. In both passive and active fingerprinting, a set of packets sent from a target host is collected by the attacker; then, these packets are compared with a range of known OS signatures. If any signature is matched, the OS type can be obtained.

In fingerprinting attacks, a vital assumption is made by an attacker that the fingerprint of the target host is static. In fact, the static nature of the network gives the attacker a large advantage because they have relatively unlimited time and methods to explore the target. However, it is difficult for the defender to deal with every exploration because unknown attack methods always exist. However, if the fingerprint of a host is changed over time, an attacker will observe a dynamic fingerprint while the exploration space [6] of the attacker is enlarged. Thus, the attacker cannot accurately determine the target host OS. This is the idea behind moving target defense (MTD) [7–10]. MTD has recently been proposed to eliminate the asymmetric advantage of attackers, which shifts the attack surface [11] of the system to achieve an unpredictable network, effectively reducing the vulnerability exposure.

In this paper, a fingerprint hopping method (FPH) is proposed using MTD to enhance the host's ability to defend against fingerprinting attacks. First, a terminal-transparent

architecture for FPH is constructed based on software-defined networks (SDN) [12]. Second, the interaction of a fingerprinting attack and defense is modeled as a signal game with consideration given to both active and passive fingerprinting. The equilibriums of the game are analyzed to obtain an optimal defense strategy. Third, an algorithm of selecting defense strategy is described. Experiments show that FPH can effectively defend against fingerprinting attacks.

2. Related Work

Honeypots are a traditional approach to defend against attackers that are attempting to fingerprint intranet hosts. Researchers use honeypots as a mechanism to deceive fingerprinting attackers and provide activity logs to defend against attacks. La et al. [13] proposed a game-based method for honeypot-enabled networks to defend against sophisticated attackers who attempt to deceive the defender by using different types of attacks. The equilibriums of both single and repeated games are analyzed to determine the optimal defense strategy. To make the best use of honeypot resources, HoneyMix [14], an SDN-based intelligent honeynet, has been proposed by Han et al., which takes advantage of SDN to achieve fine-grained flow control. HoneyMix forwards suspicious packets to a set of honeypots and replies to the attacker with the most desirable responses. Fan et al. [15] proposed a flexible general platform that supports deploying various types of honeypots. A dynamic configuration is used in virtual honeypot management to adapt to the changing network environment. However, these methods can only address attackers who try to communicate with a honeypot. If an attacker fingerprints the target host directly, these defense mechanisms will lose effectiveness. FPH is able to tackle the situation where an attacker has obtained the IP addresses of the target hosts and launched fingerprinting attack directly to the target hosts.

Packet scrubbing is a straightforward method that is used to avoid revealing intranet host information. Smart et al. [16] proposed a fingerprint scrubber to defend fingerprinting attacks. The scrubber removes identifiable information from all the packets in communication to prevent identification of the target host OS. However, this exhaustive defense method degenerates the communication performance because fingerprint scrubber modifies various fields in the packet header that are critical to performance and this method treats a benign sender and an attacker in the same way. Different from fingerprint scrubber, FPH tries to differentiate benign sender from attacker and utilizes game theory to get an optimal defense strategy to reduce the defense cost. Deceiving approaches are another way to defend against fingerprint attackers. These approaches distort the view of the attackers regarding the target host. Rahman et al. [17] proposed a game-theory approach named DeceiveGame to deceive fingerprinting attackers. Two types of senders are considered in this method, and the optimal strategy is obtained based on the equilibrium of the game. DeceiveGame scrubs fingerprint in outgoing packets and some fields of packets are randomized. However, FPH transforms the fingerprint in the packets into another fingerprint so that the attacker will misjudge the OS of target host, which can steer attackers away from the target hosts or deceive them to launch an invalid attack. Albanese et al. [18] proposed a graph-based approach to deceive attackers who are performing target host fingerprinting. The fingerprint of the host changes by manipulating the responses of the attacker's probes, but in a static way. The fingerprint of a host is transformed to another one. Different from this method, FPH hops the fingerprints in real time to achieve a dynamic host fingerprint and brings more obfuscation to the fingerprinting attacks.

MTD-based defense methods change the system surface to increase the cost and complexity for the attackers. Fulp et al. [19] proposed a resilient configuration management that changes the configuration of the host based on an evolutionary algorithm. The vulnerability exposure is reduced, and the cost to the attacker increases. Unlike this method, instead of changing the terminal configuration, FPH transparently diversifies the responses to suspicious traffic, which can be easily deployed. Wang and Wu [20] proposed a sniffer reflector based on SDN to defend against reconnaissance attacks. This method builds a shadow network for suspicious traffic to obfuscate the attacker's view of the network. However, if a false alarm appears, normal communications will be influenced. FPH changes the packet fingerprint instead of its destination, which ensures normal communications even if a false alarm appears.

OF-RHM [21], a flexible IP hopping method based on SDN, has been proposed by Jafarian et al., which can randomly mutate an IP address to defend against scanning attacks. J. Sun and K. Sun [22] proposed a seamless IP randomization method to mitigate reconnaissance attacks. The host IP addresses mutate randomly to confuse attackers; then, legitimate communications are migrated seamlessly and kept alive. However, the above two methods lose their power when it comes to fingerprinting attack, as OS information still can leak even if real IP cannot be sniffed. FPH is able to change the external view of the OSes and limit the information obtained by an attacker. RRM [23, 24], a route hopping method, has been proposed by Duan et al., which can protect 90% of traffic flow from being sniffed. Instead of hopping routes in the network, FPH tries to change some attributes of the packets of outgoing traffic to defense fingerprinting attackers. Badishi et al. proposed a random port hopping method [25], which can repel DoS attacks by changing the communication port in an unpredictable way. This method randomizes the port of a packet, but attacker still can analyze the fingerprint through the IP header or TCP options. However, FPH hops the fingerprint in packets dynamically to confuse the attacker. DHC has been proposed in the literature [26], which changes multiple network configurations, including end information and the route, to resist sniffer attackers. But the fingerprint of hosts is not removed. FPH changes multiple fields in the packets and manipulates the attacker's view of the target host's OS.

Similar to proposed work, Kampanakis et al. [27] proposed a novel SDN-based OS hiding method against fingerprinting attack. Their method forges OS fingerprints to confuse attackers based on MTD technique. TCP sequence numbers as well as payload pattern in TCP, UDP, and ICMP

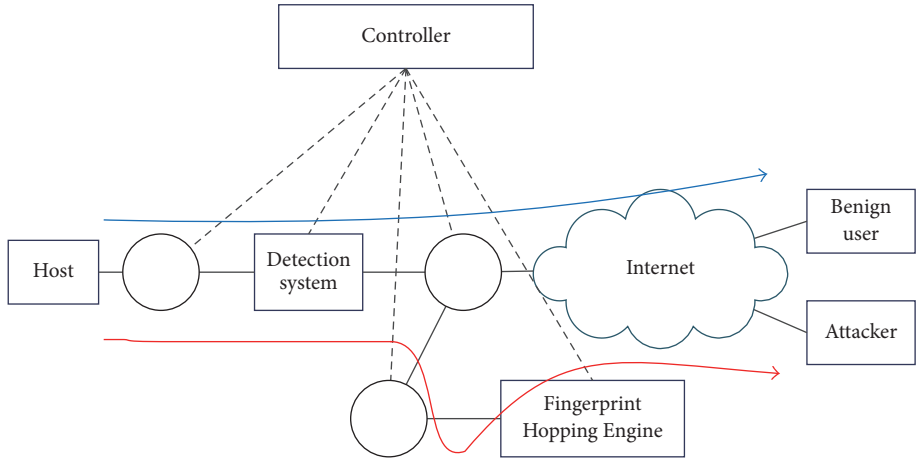

Figure 1: FPH architecture. (The blue line indicates the outgoing traffic route in a benign communication; the red line indicates the outgoing traffic route in a suspicious communication.)

are randomized for hiding the OS information. If illegitimate traffic is detected, a random sequence number or payload will be generated to respond to the opponent and a large overhead will be introduced to the attacker. However, a well-elaborated fingerprint hopping strategy may defend the fingerprinting attack with minor defense cost. FPH analyzes the fingerprinting attack and defense game and further provides optimal fingerprint hopping strategies for different situations based on the equilibriums of the game. Then, a strategy selection algorithm is proposed to maximize defense utility.

3. System Description of FPH

FPH monitors the traffic of each connection and identifies potential fingerprinting attackers based on the traffic pattern. If a communication is considered to have a fingerprinting behavior, the outgoing traffic of the communication will be rerouted and modified to hop their fingerprints. A flexible network configuration is needed to achieve traffic rerouting without communication interruption.

The powerful network management of SDN is used to construct a FPH system, as shown in Figure 1. It is a system that is transparent to the terminals because no terminal modifications are needed. The Controller, IDS, and Fingerprint Hopping Engine are the three main components of FPH. As the manager of the intranet, the Controller takes charge of route management. If fingerprint hopping is needed, the Controller generates corresponding flow entries and installs them on the switches to deliver packets from the protected host to the Fingerprint Hopping Engine. The IDS monitors the network traffic and detects the fingerprinting probes during communication. If any fingerprinting probes are detected, the IDS will inform the Controller to develop a strategy. The Fingerprint Hopping Engine is in charge of modifying fingerprints in response to fingerprinting probes and sending the response packets back to the network. It changes fingerprint in packets by modifying several fields in the packets, such as order of TCP options, the pattern of initial sequence numbers, the initial window size, TTL value, and some application layer protocol fields.

FPH can detect suspicious packets from the Internet and hop the fingerprints of responses when suspicious packets appear. However, some benign communications also have a small number of packets that can be detected as suspicious. If FPH hops fingerprints for all packets in these communications, a heavy load will be placed on the Fingerprint Hopping Engine and a large delay will be introduced into these benign communications. Furthermore, with the knowledge of the strategy of the defender, the fingerprinting attacker will hide his identity to avoid detection by FPH. A sophisticated fingerprinting attacker will try to remain "normal" as a benign user to deceive the defender and carefully conduct fingerprinting to maximize the collection of fingerprint information. However, the defender hopes to allow only benign users to access the host on the intranet and randomly hop the fingerprints of the outgoing packets of any suspicious communication within an appropriate cost. To model this interaction, a fingerprinting attacker and defender game is formulated in the next section.

4. Fingerprint Attack and Defense Game

When an attacker fingerprints a remote host, two modes can be adopted by the attacker. One mode is the "Normal" mode through which the attacker communicates with the target host in a normal way. In the "Normal" mode, the attacker can obtain limited information about the target host, but the attacker is hard to be detected by the defender because he communicates with the target host as a benign user. On the other hand, the other mode is the "Suspicious" model. In this case, the attacker sends suspicious probes to the target host and much more information about the target OS can be obtained. However, the "Suspicious" mode is much more likely to be detected by the defender because it is one of the attack patterns.

Multiple attackers may present in the network at the same time. For each of them, the interaction with the defender of

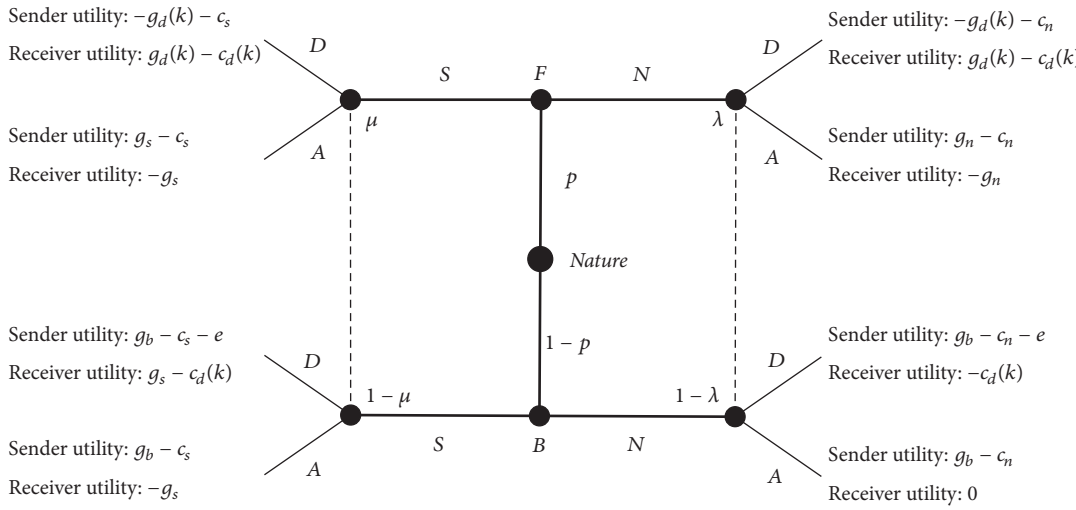

Figure 2: Extensive form of the fingerprinting attack and defense game.

the network can be modeled as a game. Here, we analyze each attacker-defender pair separately. There are two sender types, fingerprinting attacker and benign user. The fingerprint attacker tries to fingerprint the target host, and the benign user communicates with the host normally on the intranet. The two types of senders can communicate in two modes, Normal and Suspicious. The receiver is a defender of the intranet who monitors the network traffic and develops the defense strategy. When a fingerprinting behavior appears, the "defense" strategy is adopted to randomly hop the fingerprint of the protected host. Otherwise, an "Abstain" strategy is adopted to allow the sender to communicate with the intranet hosts.

4.1. Game Model. The interaction between the sender and receiver can be formulated as a game. Known from the interaction of the two players, the sender acts first (Normal or Suspicious); then, the receiver can observe the action and take action accordingly. Therefore, the game is a dynamic game. Moreover, the type of sender is private information to the receiver, and the game is an incomplete information game. By observing the actions of the sender, the receiver can infer the type of sender and selects an action (Defense or Abstain) based on the information regarding the sender type. This fingerprinting attack and defense can be modeled as a signaling game, and the definition is as follows.

Definition 1. The fingerprinting attack and defense game is a 5-tuple $(\Omega, \Theta, \Sigma, P, U)$.

$\Omega = \{\text{Sender, Receive}\}$ denotes the player set and consists of one sender and one receiver in the game.

$\Theta = \{F, B\}$ is the type space of the sender, where F denotes the fingerprinting attacker and B denotes the benign user.

$\Sigma = (M_S \times M_S) \times (A_R \times A_R)$ is the strategy combination space of the game. $M_S = \{S, N\}$ is the signal space of the sender, where S and N denote the Suspicious mode and Normal mode, respectively. $M_S \times M_S$ is the strategy space of the sender. For $(m_S, m'_S) \in M_S \times M_S$, m_S and m'_S are the signals for F-type and B-type senders, respectively. $A_R = \{D, A\}$ is the action space of the receiver, where D and A denote the Defense and Abstain actions, respectively. $A_R \times A_R$ is the strategy space of the receiver. For $(a_R, a'_R) \in A_R \times A_R$, a_R is the action for the signal S from the sender and a'_R is the action for signal N.

$P : \Theta \mapsto [0, 1]$ is the prior probability over the sender types or the belief of the defender regarding its opponent. $P = (p, 1 - p)$, where $p = \mathbb{P}(F)$, $1 - p = \mathbb{P}(B)$.

$U = (u_S, u_R)$. $u_S : \Theta \times M_S \times A_R \mapsto \mathbb{R}$ is the utility function of the sender, and $u_R : \Theta \times M_S \times A_R \mapsto \mathbb{R}$ is the utility function of the receiver.

The fingerprinting attack and defense game can be represented as the extensive form shown in Figure 2, where each branch represents a special situation with one type of sender. The nodes connected by the dotted line constitute an information set in which the defender cannot distinguish the nodes because the sender type is unknown. As seen in Figure 2, there are two information sets in this game. The left set is indicated as the S information set, and the right set is indicated as the N information set.

When the attacker fingerprints a host with probes, if the defender takes action A, the host OS information will be exposed. The attacker can benefit from this process under the risk of being detected by the defender. For the fingerprinting attacker, g_s and g_n are introduced to denote the benefit of the attacker given the signals S and N, respectively. c_s and c_n denote the cost of the attacker given the two signals, which is caused by the risk. Note that, for the attacker, a suspicious probe will obtain much more information than a normal probe and also increase the risk correspondingly. Therefore, it is assumed that $g_s > g_n$ and $c_s > c_n$. Considering a zero sum model, the more the attacker benefits (e.g., g_s), the more losses the defender suffers (e.g., $-g_s$).

For the defender, it is assumed that the fingerprint hopping space of the protected host is Ξ and the size of the hopping space is $k = |\Xi|$, which means that the defender can randomly select one of k different OS fingerprints to answer

the attacker. If the defender replies to a fingerprinting attacker with Ξ, he will receive benefit $g_d(k)$ and pay cost $c_d(k)$. $g_d(k)$ and $c_d(k)$ increase with k because if the fingerprint space is larger, it will be more difficult for the attacker to discover the real fingerprint of the target host and the defender will take more resources, that is monotone increase function.

The utilities of both players in every situation are modeled as Utility = Benefit − Cost. In Figure 2, when the type of sender is F and (S, D) is played by the sender and receiver, the cost of the sender is c_s and the benefit is $-g_d(k)$, which is caused by the hopping fingerprint defense. Therefore, the utility of the sender is $-g_d(k) - c_s$. The receiver benefits $g_d(k)$, and the cost of the receiver is $c_d(k)$; therefore, the utility of the receiver is $g_d(k) - c_d(k)$ ($g_d(k) - c_d(k) > 0$). When the type of the sender is F and (N, D) is played by the sender and receiver, the sender will obtain a hopping fingerprint, so he will obtain benefit $-g_d(k)$ and cost c_n. The receiver benefits $g_d(k)$ and cost $c_d(k)$; therefore, the utility of the receiver is $g_d(k) - c_d(k)$. When the type of sender is B and (S, A) is played by the sender and receiver, the sender achieves benefit g_b ($g_b > 0$) because the benign user communicates with the target host successfully. In this case, the benefit of the receiver is $-g_s$ because the receiver responds to the sender with real fingerprint information that can be sniffed by a passive fingerprinting attacker. When the type of sender is B and (S, D) is played by the sender and receiver, the cost of the sender is $c_s + e$, where cost e is caused by the delay addition from hopping fingerprints. The benefit of the receiver is g_s because fingerprint information leakage is prevented using the hopping fingerprint. It is assumed that the utility of the receiver is $g_s - c_d(k) > 0$. When the type of sender is B and (N, A) is played by the sender and receiver, the utility of receiver is assumed to be 0 because the defender neither prevents fingerprint leakage nor takes a defensive measure. Other situations are easy to understand.

4.2. Equilibriums Analysis. As mentioned previously, the interaction between a fingerprinting attack and its defense has been modeled as a signaling game, where Perfect Bayesian Equilibrium (PBE) [28] is used to predict the outcome of the game. PBE describes the complete course of action of both players, which is an optimal strategy for all of the players of the game. None of the players can obtain a higher utility if they deviate from the PBE strategy. In the fingerprinting attack and defense game, a PBE is defined as a strategy combination; that is, PBE $\triangleq ((m_S, m_S'), (a_R, a_R')) \in \sum$. (m_S, m_S') describes the signals for both types of senders and (a_R, a_R') describes the actions of the receiver as responses to the two potential signals sent by the sender. When the receiver observes a signal from the sender, the posterior probability of the sender type can be computed based on Bayes' rule. In the fingerprinting attack and defense game, the posterior probabilities are defined as (μ, λ), as shown in Figure 2, where

$$\begin{aligned} \mu &= \mathbb{P}(F \mid S), \\ 1 - \mu &= \mathbb{P}(B \mid S), \\ \lambda &= \mathbb{P}(F \mid N), \\ 1 - \lambda &= \mathbb{P}(B \mid N). \end{aligned} \tag{1}$$

In the signal game, a pooling equilibrium means that both types of senders send the same signal. A separating equilibrium is a strategy in which different types of senders send different signals. In this section, all of the pooling equilibriums and separating equilibriums are analyzed for the fingerprinting attack and defense game.

4.2.1. Pooling PBE. There are two pooling strategies for the sender: (N, N) and (S, S). The pooling strategy (N, N) is examined first.

Theorem 2. *The fingerprinting attack and defense game has a pooling PBE* $((N, N)(D, D))$ *if* $p \geq c_d(k)/(g_d(k) + g_n)$.

Proof. The sender pooling strategy (N, N) means that the sender plays N in the game regardless of his type. Given the sender strategy (N, N), the information set N in Figure 2 is reached and the posterior probability about sender type can be calculated by Bayes' rule, as shown in

$$\lambda = \mathbb{P}(F \mid N) = \frac{\mathbb{P}(F)}{\mathbb{P}(F) + \mathbb{P}(B)} = \frac{p}{p + 1 - p} = p. \tag{2}$$

Using this posterior probability, the expected utility of the two actions of the receiver are shown in the following.

$$\begin{aligned} &\text{For action } D,\ E_{u_R}(D \mid N) \\ &= \mathbb{P}(F \mid N) \cdot u_R(F, N, D) + \mathbb{P}(B \mid N) \\ &\quad \cdot u_R(B, N, D) \\ &= p\left(g_d(k) - c_d(k)\right) + (1 - p)\left(-c_d(k)\right) \\ &= pg_d(k) - c_d(k) \end{aligned} \tag{3}$$

$$\begin{aligned} &\text{For action } A,\ E_{u_R}(A \mid N) \\ &= \mathbb{P}(F \mid N) \cdot u_R(F, N, A) + \mathbb{P}(B \mid N) \\ &\quad \cdot u_R(B, N, A) = p(-g_n) + (1 - p) \cdot 0 = -pg_n. \end{aligned} \tag{4}$$

If $p \geq c_d(k)/(g_d(k) + g_n)$, $E_{u_R}(D \mid N) \geq E_{u_R}(A \mid N)$ can be obtained. In other words, D is the best response for the receiver given signal N. Thus, the utility of the sender is shown in the following.

$$\begin{aligned} &\text{For the } F\text{-type sender},\ u_S(F, N, D) = -g_d(k) - c_n \\ &\text{For the } B\text{-type sender},\ u_S(B, N, D) = g_b - c_n - e. \end{aligned} \tag{5}$$

To ensure that the sender has no intention to deviate from signal N, we verified whether S can provide higher utility for a sender of any type. If S is the sender signal, the information set F in Figure 2 will be reached. The receiver observes the signal

S and the expected utilities of his two responses are shown as follows.

For action D, $E_{u_R}(D \mid S)$

$$= \mu\left(g_d(k) - c_d(k)\right) + (1-\mu)\left(g_s - c_d(k)\right) \quad (6)$$

For action A, $E_{u_R}(A \mid S) = \mu(-g_s) + (1-\mu)(-g_s)$.

D is better receiver response because $E_{u_R}(D \mid S) > 0 > E_{u_R}(A \mid S)$. Therefore, the utilities of senders of both types are shown in the following.

$$\begin{aligned} &\text{For the } F\text{-type sender, } u_S(F,S,D) = -g_d(k) - c_s \\ &\text{For the } B\text{-type sender, } u_S(B,S,D) = g_b - c_s - e. \end{aligned} \quad (7)$$

From (5) and (7), $u_S(F,N,D) > u_S(F,S,D)$ and $u_S(B,N,D) > u_S(B,S,D)$ can be obtained, which mean that the signal N can provide higher utility for both sender types. Therefore, the sender will not deviate from N; that is, $((N,N)(D,D))$ is a pooling PBE of the game if $p \geq c_d(k)/(g_d(k)+g_n)$. □

Theorem 3. *The fingerprinting attack and defense game has a pooling PBE* $((N,N)(D,A))$ *if* $p < c_d(k)/(g_d(k)+g_n)$.

Using the same process, Theorem 3 can be proved. Theorems 2 and 3 show that the optimal strategy for a fingerprinting attacker is to appear normal, as a benign user. If the prior probability p is larger than a certain threshold, the defender will hop fingerprints for every packet, regardless of the signal of the opponent. Otherwise, the defender will play D for signal S and play A for signal N. It can also be proved that the pooling strategy (S,S) is not a part of PBE using the same process, and the details are omitted.

4.2.2. Separating PBE

Theorem 4. *The fingerprinting attack and defense game has no separating PBE.*

Proof. There are two possible separating strategies for the sender in this game: (S,N) and (N,S). (S,N) will be first discussed below.

Assuming that (S,N) is the strategy for the sender or that the F-type sender only sends signal S and the B-type sender only sends signal N, the utility of the sender is discussed as follows.

(1) If the sender is F-type, S is the signal of the sender according to the separating strategy. In this case, if the receiver plays D, he will obtain utility $g_d(k)-c_d(k)$. Otherwise, if the receiver plays A, he will obtain utility $-g_s$. $g_d(k)-c_d(k) > 0 > -g_s$, so the optimal action for the receiver is D; thus, the utility of the F-type sender is $-g_d(k) - c_s$.

(2) If the sender is B-type, N is the signal of the sender according to the separating strategy. In this case, if the receiver plays D, he will obtain utility $-c_d(k)$. Otherwise, if the receiver plays A, he will obtain utility 0. Obviously, A is the optimal action for the receiver because $-c_d(k) < 0$. Thus, the utility of the sender is $g_b - c_n$.

Given the receiver strategy (D,A), it is verified whether the sender will deviate from the separating strategy (S,N). If the F-type sender deviates from S to N, A is the receiver response and the sender will obtain utility $g_n - c_n$, which is larger than the utility when he plays S. Thus, the sender will deviate from signal S to N. Therefore, the separating strategy (S,N) is not part of a PBE.

For the other separating strategy for the sender (N,S), the same conclusion can be obtained using a similar process and the details are omitted.

In conclusion, the fingerprinting attack and defense game has no separating PBE. □

4.3. Belief Model. In order to facilitate the analysis, the conclusions of Theorems 2 and 3 are obtained under an ideal condition that both the false positive rate (FP) and false negative rate (FN) of IDS are zero. In reality, small parts of suspicious probes cannot be detected by the IDS (FN > 0). It is also possible that a benign user can send a few suspicious packets in some special situations. With this knowledge, the fingerprinting attacker will send some suspicious probes to obtain more information about the target host OS. When the defender identifies suspicious packets from a sender, the belief of the defender about the sender type will be updated. Function $p(t)$ is defined as the belief of the defender instead of the constant p when t suspicious packets are received. Similar to the literature [17], $p(t)$ is formalized as

$$p(t) = \min\left(1, \frac{e^{(a_0+\varphi(t))/G} - 1}{e-1}\right). \quad (8)$$

In (8), a_0 is the initial value when no suspicious packet is detected. A larger a_0 indicates that the sender is more likely to be a fingerprinting attacker. G denotes the total fingerprint information obtained by a sender. $\varphi(t)$ is the fingerprint information gained for the sender in the communication, which can be calculated by (9), where r_i is the fingerprint information gain for the ith suspicious packet [17]. θ $(0 < \theta \leq 1)$ represents the ratio of fingerprint information that can be reconnoitered by probes detected by IDS to that which can be reconnoitered by all probes sent by the attacker. It can be estimated by repeated tests on IDS using fingerprinting tools, such as Nmap. When a part of probes is not detected (FN > 0), some fingerprint information is leaked; that is $\theta < 1$. Note that $\varphi(0) = 0$.

$$\varphi(t) = \frac{1}{\theta}\sum_{i=1}^{t} r_i. \quad (9)$$

The exponential function is chosen as the belief function so that a unit increase of fingerprint information obtained by the sender leads to higher increase of suspiciousness with the increase of already obtained fingerprint information.

4.4. Fingerprint Hopping Space. Assuming that Ξ is the fingerprint hopping space for a protected host h and $k = |\Xi|$, fingerprint$(h) \in \Xi$, where fingerprint(h) is the real fingerprint of host h. In other words, the fingerprint hopping

Input: $t, r_1, r_2, \ldots, r_t$
Output: *Strategy*
StrategySelect
(01) $p^* = c_d(k_m)/(g_d(k_m) + g_n)$
(02) $\varphi(t) = 0$
(03) **while** communication is going on
(04) **if** a new suspicious packet is detected by IDS
(05) $\varphi(t) = (1/\theta) \sum_{i=1}^{t} r_i$
(06) Get $p(t)$ using Eq. (8)
(07) **if** $p(t) \geq p^*$
(08) Select (D, D) as the strategy of the defender
(09) Get $k = \widetilde{k}_o$ using Eq. (14)
(10) Set up the strategy on the IDS and Fingerprint Hopping Engine
(11) **else**
(12) Select (D, A) as the strategy of the defender
(13) $k = k_m$
(14) Set up the strategy on the IDS and Fingerprint Hopping Engine
(15) **end while**
(16) **return**

Algorithm 1: Strategy selection algorithm.

space of h contains the real fingerprint of h because normal communication with h has exposed a part of its fingerprint. $g_d(k)$ and $c_d(k)$ are the benefit and cost of the defender, respectively, when the hopping space size is k. $g_d(k)$ and $c_d(k)$ are calculated by (10) and (11), respectively, where $u > 1$, $\beta > 0$, $k \in \mathbb{Z}^+$.

$$g_d(k) = \alpha \log_u k \tag{10}$$

$$c_d(k) = \beta k - \beta. \tag{11}$$

A logarithmic function is considered for $g_d(k)$ because the defender will benefit less with unit increase of k when the hopping space size is already large, as the addition of confusion to the attacker is less. Furthermore, $g_d(1) = 0$ should hold, which indicates that there should be no benefit for the defender if the hopping space size is 1; that is, $\Xi = \{\text{fingerprint}(h)\}$. Therefore, (10) is able to describe the property of the defender's benefit with respect to hopping space size. Other types of functions, such as exponential function and linear function, cannot reflect the relationship between defender's benefit and hopping space size. The cost function $c_d(k)$ reflects the penalty of memory consumption increased with the growth of hopping space size, which is defined as linear function, indicating fixed growth rate of hopping cost regardless of hopping space size. The defender's cost should be zero when the hopping space size is 1; that is, $c_d(1) = 0$. Other functions cannot describe the fixed growth rate of the hopping cost with the size of hopping space. As mentioned previously, when $p \geq c_d(k)/(g_d(k) + g_n)$, $((N, N)(D, D))$ is the equilibrium solution of the fingerprinting attack and defense game, and the expected utility of defender is shown in (3). Combined with (3), (10), and (11), (12) can be obtained.

$$E_{u_R} = p g_d(k) - c_d(k) = p\alpha \log_u k - (\beta k - \beta). \tag{12}$$

If k is very small, the probability of successfully deducing the correct fingerprint by the attacker will be high; however, if k is very large, the defender must bear a large defense cost. Thus, the defender will decide the value of k to maximize his expected utility. Equation (13) is obtained by deriving E_{u_R} with respect to k.

$$E'_{u_R} = \frac{p\alpha}{k \ln u} - \beta. \tag{13}$$

When E'_{u_R} is zero, the maximum expected utility is found. Thus k_o can be obtained, as shown in (14). In practical application, $\widetilde{k}_o$ is chosen as the optimal value shown in (15), where k_m is the minimum size of the fingerprint hopping space.

$$k_o = \frac{p\alpha}{\beta \ln u} \tag{14}$$

$$\widetilde{k}_o = \max(k_m, \lceil k_o \rceil). \tag{15}$$

5. Strategy Selection Algorithm

With the updated belief, the defender should adjust his strategy to maximize his utility. A strategy selection algorithm is proposed to find the optimal strategy, as shown in Algorithm 1. In the algorithm, the belief threshold p^* is found with the initial size of the fingerprint hopping space k_m. When the IDS identifies a suspicious packet, the belief of the defender about the sender type will be updated. If the belief is smaller than threshold p^*, strategy (D, A) will be played by the defender. Otherwise, (D, D) will be played.

6. FPH Design

A prototype system of FPH is designed based on SDN, as shown in Figure 3, which consists of the following three components: the Controller, IDS, and Fingerprint Hopping

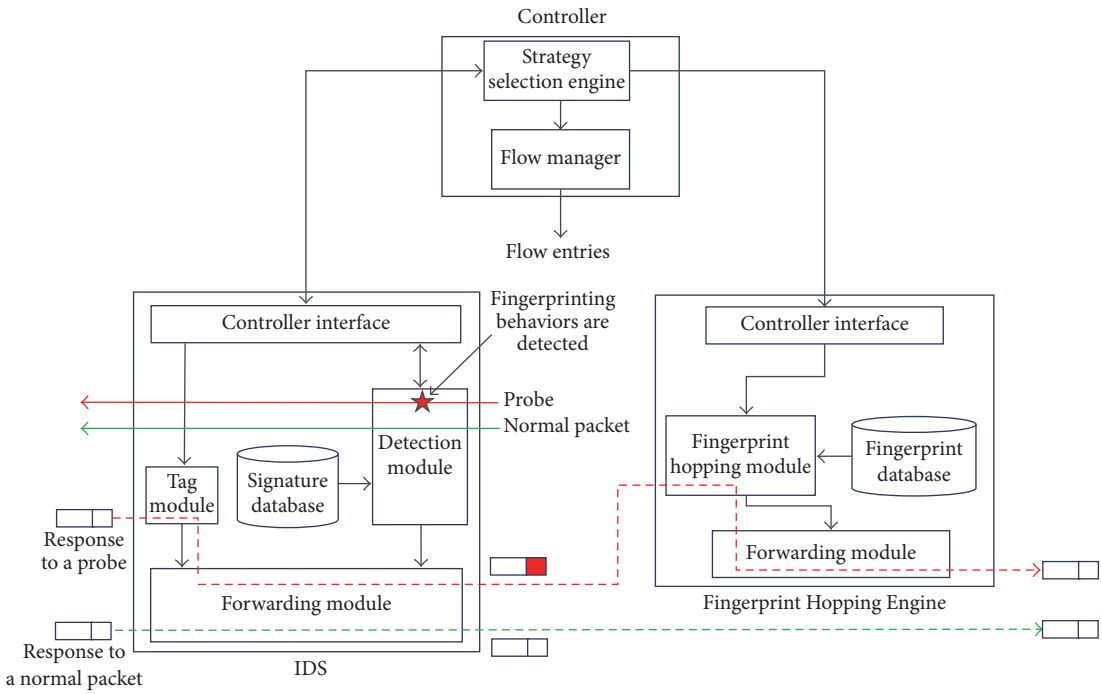

Figure 3: FPH design.

Engine. The green solid line and red solid line denote the paths of a normal packet and a fingerprinting probe, respectively. The green dash line and red dash line denote the paths of the responses of a normal packet and a fingerprinting probe, respectively. IDS monitors the packets in the communication. The Detection Module of IDS detects the fingerprinting behavior based on a Signature Database, which can be built through collecting the probe signatures of fingerprinting tools, such as Nmap. When a packet arrives, IDS will match the packet with the signatures in the database. If no signature is matched, the outgoing packets will be sent to network without modification. Otherwise, if any signature is matched, IDS will report to the Controller through the Controller Interface. When the defender strategizes to hop the fingerprint of a packet, the response packet will be tagged by the Tag Module of IDS. Then the tagged packet (red rectangle in Figure 3) will be forwarded to the network through Forwarding Module.

With the report message sent by the IDS, the Controller calculates the belief about the sender type and makes a strategy. If fingerprinting behaviors are detected, the Controller will set up flow entries to the Openflow switches through the Flow Manager to deliver the tagged response packets to the Fingerprint Hopping Engine. The Fingerprint Hopping Engine changes the fingerprint of these packets based on the size of the fingerprint hopping space informed by the Controller. Finally, the packets with the hopping fingerprints will be sent back to the network through the Forwarding Module of the Fingerprint Hopping Engine and the tag will be deleted.

To reroute the responses of suspicious packets, the tagging technique [29] is used to mark these responses. If the defender takes the Defense action, the IDS will be informed to add a tag to the responses of these suspicious packets and related flow entries will be installed on the switches to forward the packets with this tag to the Fingerprint Hopping Engine. The outgoing traffic routes of a protected host for a fingerprinting attacker and benign user are shown in Figure 4, in which the tagged packets are marked in red.

7. Experiments and Analysis

In this section, the security and performance of FPH are evaluated. The topology of the network, as shown in Figure 1, is constructed using Mininet [30] with a benign user, a fingerprinting attacker, and a target host. Openflow 1.0 [31] is applied and POX [32] is used as the Controller. In our experiments, all the evaluation examples are done on a machine with a 2.53 GHz Intel Xeon and 32 G RAM 64 bits.

7.1. Performance Evaluation. When FPH adopts hopping fingerprints to a suspicious communication, the Controller will set up related flow entries on the switches to forward the outgoing packets to the Fingerprint Hopping Engine. The Controller will also inform the Fingerprint Hopping Engine about the size of the hopping space. Due to these processes, network latency will be introduced. To evaluate the network delay, FPH is deployed based on Mininet and 10 repeated tests are conducted on a fingerprinting communication and a benign communication, which are created by Nmap and FTP, respectively. The result is shown in Figure 5, where the horizontal coordinates stand for the number of the tests in the experiment. When the communication is benign, the network delay is low, as seen in the figure, because the Defense action is not taken. FPH will not cause an additional delay for the benign user because the Defense action is only taken

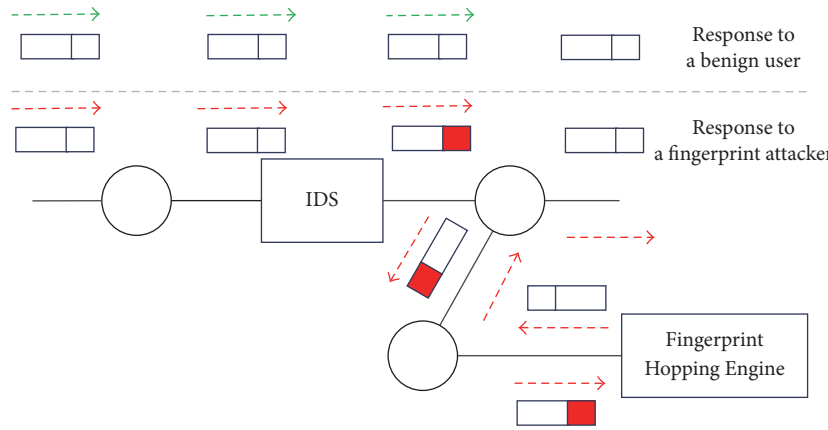

Figure 4: Fingerprint probe routes.

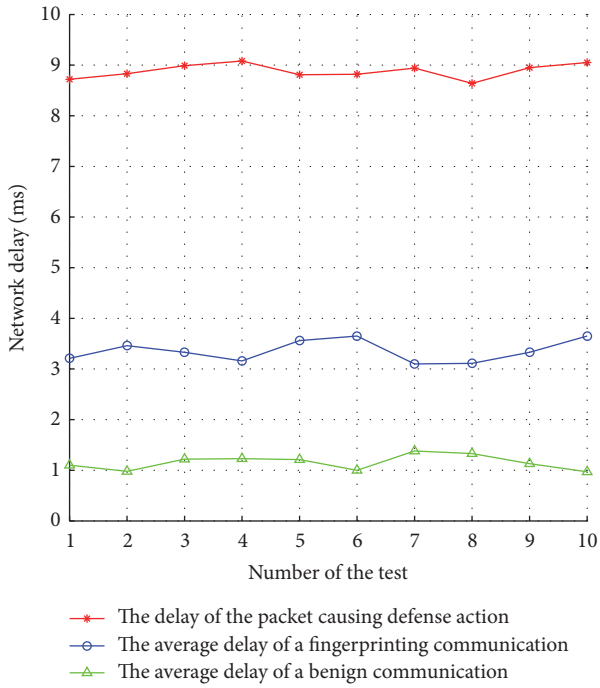

Figure 5: Network delay of FPH.

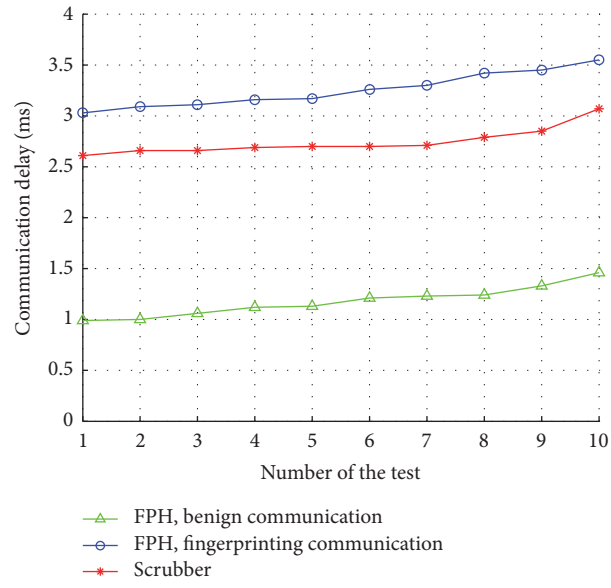

Figure 6: The comparison of communication delay between FPH and scrubber.

when the belief of the defender exceeds a certain threshold, which is unlikely to be reached by a benign user. The fingerprinting communication will cause the FPH Defense action, and the network delay of the suspicious communication will increase. A high delay is introduced for a packet that causes the Defense action because this packet has to wait in the network for the related flow entries to be set up. The average delay of the fingerprinting communication is much lower but still higher than that of the benign communication because the outgoing traffic of the fingerprinting communication will be sent to the Fingerprint Hopping Engine for modification.

Different from FPH, scrubber [16] is an exhaustive defense method, which degenerates the communication performance. In this experiment, the network delay introduced to different types of communications by FPH and scrubber are compared. We focus on the fingerprint scrubbing method described in [16], which normalizes IP type-of-service and fragment bits in the IP header. We also implement a scrubber on SDN, so that all the experiments are conducted in the same condition. The delay of each packet in the communication is collected. The results are shown in Figure 6, where the communication delays are sorted in ascending order. For the scrubber, all the packets in the communication need to be modified regardless of the type of traffic. Compared with scrubber, FPH achieves much lower communication delay when the opponent is a benign sender, because no packet modification is required in the communication, which is a time-consuming operation. Therefore, FPH can achieve lower delay for a benign communication. However, for a fingerprinting communication, the communication delay of FPH is higher than that of scrubber. The reason is that, in

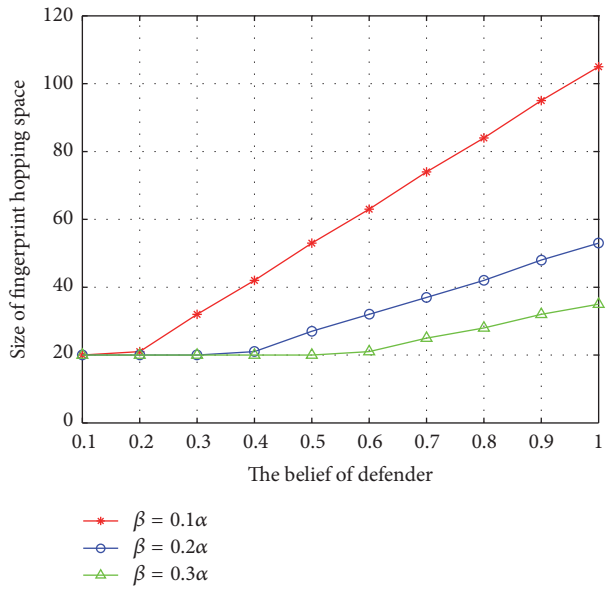

Figure 7: Fingerprint hopping space size versus defender belief.

FPH, not only does the outgoing traffic need to be modified to hop fingerprint, but also the incoming traffic needs to be monitored.

7.2. Evaluation of the Fingerprint Hopping Space. The optimal size of the hopping space, $\widetilde{k}_o$, changes with the belief of the defender. Intuitively, if the defender has a stronger belief that the opponent is a fingerprinting attacker, he will adopt a larger hopping space to confuse the attacker. Otherwise, he will adopt a smaller hopping space to save defense costs. In the experiment, α and β in (10) and (11) are set to $\beta = 0.1\alpha$, $\beta = 0.2\alpha$, and $\beta = 0.3\alpha$, with $u = 1.1$ and $k_m = 20$. Then, the optimal size of the fingerprint hopping space can be obtained using (15), as shown in Figure 7. Due to the minimum size of the fingerprint hopping space, $\widetilde{k}_o$ is a constant value when the belief value is small. When the belief value increases, $\widetilde{k}_o$ grows linearly. As seen in the figure, a smaller β produces a larger hopping space because if the fingerprint hopping costs less, the defender can adopt a larger hopping space to obtain a greater benefit from making the attacker more confused.

7.3. Security Evaluation. In this experiment, Nmap (v7.40) and p0f (v3.09b) are used as active and passive fingerprinting tools to verify the security of FPH. The target host runs on a separate VM which is connected to the network generated by Mininet. The firewall of the target host is closed and we assume that false negative rate is 0; that is, $\theta = 1$. The attacker uses Nmap to actively fingerprint the target host, and p0f is employed to passively fingerprint the target host. The commands of the two tools are as follows.

Command for Nmap: nmap -O -v target_IP

Command for p0f: p0f -i target_interface

The results of the experiment are shown in Table 1. As can be seen, the security of FPH is verified on different

Table 1: Output of the fingerprinting tools Nmap and p0f.

OS of the target host	Running FPH		No defense mechanism	
	Nmap	p0f	Nmap	p0f
Windows XP	N	N	Y	N
Windows 7	N	N	Y	Y
Windows 10	N	N	Y	YF
Ubuntu 10.10	N	NF	Y	Y
Ubuntu 11.10	N	N	Y	Y
Ubuntu 14.04	N	NF	Y	N

N: attacker fails to fingerprint the target host. NF: attacker falsely identifies the OS. Y: attacker succeeds to identify the OS. YF: attacker succeeds to identify the OS type but falsely identify the OS version.

OSes and OS versions. When no defense mechanism is adopted in the network, Nmap is able to fingerprint the target host precisely and p0f can also identify the OS of the host correctly for most cases. Windows 10 is falsely identified as Windows 7 or 8 by p0f, but the OS type is recognized correctly. Windows XP and Ubuntu 14.04 are not identified by p0f. This is because the feature database does not contain features that match the packets sent by target host. However, both the two fingerprinting tools fail to detect OS of the target host when FPH is adopted. Since the responses of the probes sent by Nmap are modified by FPH, the fingerprint observed by the attacker changes dynamically. As a result, Nmap cannot recognize the OS of target host through analyzing the responses. It also can be seen that, in some cases, p0f falsely identifies the target OS. The reason is that p0f fingerprints the target host using the attributes of single packet. FPH transforms the fingerprint in the packet into another fingerprint, so p0f misjudges the OS of target host, which will steer the attacker away from the target host or deceive them to launch an invalid attack.

8. Conclusions and Future Work

Fingerprinting is an essential step for network attacks, which enables the attacker to obtain the OS information of target host for attackers. In this paper, FPH is proposed based on SDN to provide a hopping fingerprint for attackers to resist fingerprinting attacks. Using the idea of MTD, FPH hops the fingerprint of the protected host to expand the exploration space of the attacker and disable the fingerprinting tools. The fingerprinting attack and defense game is modeled, and the equilibriums of the game are analyzed. An appropriate defense strategy is presented with sender type consideration. Experiments show that FPH can effectively defend against fingerprinting attacks. In this paper, the interactions of fingerprinting attack and defense are modeled as a series of one-shot games and the change of defender's belief is taken into consideration. However, we assume that only the defender has the knowledge of game history. In future work, a multistage game will be modeled for continuous interaction between the fingerprinting attack and defense. In addition, a more reasonable assumption that both the attacker and defender have knowledge of game history will be made

and experiments where both attacker and defender adopt strategies derived based on this history will be conducted.

Competing Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (nos. 61401512, 61379151, 61272489, and 61302159) and The National Cryptography Development Fund of China (no. MMJJ201301005).

References

[1] M. Zalewski, "The new p0f: 2.0.8," September 2006, http://lcamtuf.coredump.cx/p0f.shtml.

[2] P. Auffret, "SinFP, unification of active and passive operating system fingerprinting," *Journal in Computer Virology*, vol. 6, no. 3, pp. 197–205, 2010.

[3] G. Lyon, "Nmap: a free network mapping and security scanning tool," 2014, http://nmap.org/.

[4] O. Arkin and F. Yarochkin, "Xprobe v2.0: a fuzzy approach to remote active operating system fingerprinting," Tech. Rep., 2002.

[5] F. V. Yarochkin, O. Arkin, M. Kydyraliev, S.-Y. Dai, Y. Huang, and S.-Y. Kuo, "Xprobe2++: low volume remote network information gathering tool," in *Proceedings of the IEEE/IFIP International Conference on Dependable Systems and Networks (DSN '09)*, pp. 205–210, Lisbon, Portugal, July 2009.

[6] R. Zhuang, S. A. DeLoach, and X. Ou, "Towards a theory of moving target defense," in *Proceedings of the 1st ACM Workshop on Moving Target Defense (MTD '14)—Co-located with 21st ACM Conference on Computer and Communications Security (CCS '14)*, pp. 31–40, Scottsdale, Ariz, USA, November 2014.

[7] A. Ghosh, D. Pendarakis, and W. Sanders, "Moving target defense co-chair's report-National Cyber Leap Year Summit 2009," Tech. Rep., Federal Networking and Information Technology Research and Development (NITRD) Program, 2009.

[8] T. Cyberspace, *Strategic Plan for the Federal Cybersecurity Research and Development Program*, Executive Office of the President National Science and Technology Council, Washington, DC, USA, 2011.

[9] S. Jajodia, A. K. Ghosh, V. Swarup, C. Wang, and X. S. Wang, *Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats*, vol. 54, Springer Science & Business Media, 2011.

[10] S. Jajodia, A. K. Ghosh, V. Subrahmanian, V. Swarup, C. Wang, and X. S. Wang, *Moving Target Defense II. Application of Game Theory and Adversarial Modeling*, Advances in Information Security, Springer, Berlin, Germany, 2013.

[11] P. K. Manadhata, D. K. Kaynar, and J. M. Wing, *A Formal Model for a System's Attack Surface*, DTIC Document, 2007.

[12] N. McKeown, "Software-defined networking," *INFOCOM Keynote Talk*, vol. 17, no. 2, pp. 30–32, 2009.

[13] Q. D. La, T. Q. Quek, J. Lee, S. Jin, and H. Zhu, "Deceptive attack and defense game in honeypot-enabled networks for the internet of things," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 1025–1035, 2016.

[14] W. Han, Z. Zhao, A. Doupé, and G.-J. Ahn, "HoneyMix: toward SDN-based intelligent honeynet," in *Proceedings of the ACM International Workshop on Security in Software Defined Networks and Network Function Virtualization*, pp. 1–6, 2016.

[15] W. Fan, D. Fernández, and Z. Du, "Versatile virtual honeynet management framework," *IET Information Security*, vol. 11, no. 1, pp. 38–45, 2017.

[16] M. Smart, G. R. Malan, and F. Jahanian, "Defeating TCP/IP stack fingerprinting," in *Proceedings of the 9th USENIX Security Symposium*, Denver, Colo, USA, August 2000.

[17] M. A. Rahman, M. H. Manshaei, and E. Al-Shaer, "A game-theoretic approach for deceiving remote operating system fingerprinting," in *Proceedings of the 1st IEEE International Conference on Communications and Network Security (CNS '13)*, pp. 73–81, October 2013.

[18] M. Albanese, E. Battista, S. Jajodia, and V. Casola, "Manipulating the attacker's view of a system's attack surface," in *Proceedings of the IEEE Conference on Communications and Network Security (CNS '14)*, pp. 472–480, San Francisco, Calif, USA, October 2014.

[19] E. W. Fulp, H. D. Gage, D. J. John, M. R. McNiece, W. H. Turkett, and X. Zhou, "An evolutionary strategy for resilient cyber defense," in *Proceedings of the 58th IEEE Global Communications Conference (GLOBECOM '15)*, San Diego, Calif, USA, December 2015.

[20] L. Wang and D. Wu, "Moving target defense against network reconnaissance with software defined networking," in *Proceedings of the International Conference on Information Security*, 2016.

[21] J. H. Jafarian, E. Al-Shaer, and Q. Duan, "OpenFlow random host mutation: transparent moving target defense using software defined networking," in *Proceedings of the 1st ACM International Workshop on Hot Topics in Software Defined Networks (HotSDN '12)*, pp. 127–132, Helsinki, Finland, August 2012.

[22] J. Sun and K. Sun, "DESIR: Decoy-Enhanced Seamless IP Randomization".

[23] Q. Duan, E. Al-Shaer, and H. Jafarian, "Efficient random route mutation considering flow and network constraints," in *Proceedings of the 1st IEEE International Conference on Communications and Network Security (CNS '13)*, October 2013.

[24] J. Jafarian, E. Al-Shaer, and Q. Duan, "Formal approach for route agility against persistent attackers," in *Computer Security—ESORICS 2013*, J. Crampton, S. Jajodia, and K. Mayes, Eds., pp. 237–254, Springer, Berlin, Germany, 2013.

[25] G. Badishi, A. Herzberg, and I. Keidar, "Keeping denial-of-service attackers in the dark," *IEEE Transactions on Dependable and Secure Computing*, vol. 4, no. 3, pp. 191–204, 2007.

[26] Z. Zhao, D. Gong, B. Lu, F. Liu, and C. Zhang, "SDN-based double hopping communication against sniffer attack," *Mathematical Problems in Engineering*, vol. 2016, Article ID 8927169, 13 pages, 2016.

[27] P. Kampanakis, H. Perros, and T. Beyene, "SDN-based solutions for Moving Target Defense network protection," in *Proceedings of the IEEE 15th International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM '14)*, pp. 1–6, IEEE, Sydney, Australia, 2014.

[28] R. Gibbons, *Game Theory for Applied Economists*, Princeton University Press, Princeton, NJ, USA, 1992.

[29] Z. A. Qazi, C.-C. Tu, L. Chiang, R. Miao, V. Sekar, and M. Yu, "SIMPLE-fying middlebox policy enforcement using SDN,"

in *Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM (SIGCOMM '13)*, pp. 27–38, Hong Kong, China, August 2013.

[30] B. Lantz, B. Heller, and N. McKeown, "A network in a laptop: rapid prototyping for software-defined networks," in *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks (HotNets '10)*, Monterey, Calif, USA, October 2010.

[31] N. McKeown, T. Anderson, H. Balakrishnan et al., "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[32] M. McCauley, "About pox," 2013, http://www.noxrepo.org.