Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2016, Article ID 8927169, 13 pages
http://dx.doi.org/10.1155/2016/8927169

Research Article

SDN-Based Double Hopping Communication against Sniffer Attack

Zheng Zhao,[1] Daofu Gong,[1] Bin Lu,[1] Fenlin Liu,[1] and Chuanhao Zhang[2,3]

[1]*Zhengzhou Science and Technology Institute, Zhengzhou 450002, China*
[2]*Railway Police College, Zhengzhou 450002, China*
[3]*National Digital Switching System Engineering & Technological R&D Center, Zhengzhou 450002, China*

Correspondence should be addressed to Zheng Zhao; diyigemsn@hotmail.com

Received 7 September 2015; Revised 6 December 2015; Accepted 8 December 2015

Academic Editor: Oleg V. Gendelman

Sniffer attack has been a severe threat to network communication security. Traditional network usually uses static network configuration, which provides convenience to sniffer attack. In this paper, an SDN-based double hopping communication (DHC) approach is proposed to solve this problem. In DHC, ends in communication packets as well as the routing paths are changed dynamically. Therefore, the traffic will be distributed to multiple flows and transmitted along different paths. Moreover, the data from multiple users will be mixed, bringing difficulty for attackers in obtaining and recovering the communication data, so that sniffer attack will be prevented effectively. It is concluded that DHC is able to increase the overhead of sniffer attack, as well as the difficulty of communication data recovery.

1. Introduction

Sniffer attack is a serious matter for network communication security. Sniffer attack is one of the most popular ways used by attackers, which captures and analyzes network communication data. Sniffer attackers are able to eavesdrop communication data from network nodes or links, monitor network status, and steal sensitive data such as usernames and passwords. However, the static network configurations in traditional network provide convenience for sniffer attack. For instance, static ends and route configurations make it easy for attackers to obtain and analyze communication data.

Communication encryption is a traditional approach to preventing sniffer attack. The communication data is encrypted during transmission, making it difficult for attackers to crack the information. However, there are still some limitations in practical applications. Firstly, encryption protocol should be supported by both communicating sides or communication would fail. Secondly, a large number of popular protocols, such as HTTP, FTP, Telnet, and SMTP, do not apply encryption, which causes serious security risk to communication based on these protocols. Thirdly, security flaws exist in some encryption protocols, by which attackers may crack communication data.

Moving target defense (MTD) [1–4], a recently proposed technology, uses dynamicity to enhance communication security. The network configuration is dynamically changed to deceive attackers [5, 6], avoid attacks [7–9], and defend against attacks [10, 11]. However, potential attacks still exist even if single network configuration is changed [5]. Changes of multiple network configurations can enhance the dynamicity of the network and further improve network security.

Collaborative changes of multiple network configurations put forward higher requirements on capabilities of networks management. Distributed control is adopted in traditional IP network, in which the routing table configuration relies on routing protocols. In this paradigm, serious consequences, such as service interruptions and routing inflation, can appear due to the changing network configuration [9]. And it is hard for traditional network to change multiple network configuration collaboratively. For example, it is difficult for MPLS, a high-speed networking technique used in traditional network, to implement dynamic resources changes due to the lack of a global view and flexible resource allocation

[12]. Dynamic transformation of host IP configuration is attempted to be realized in traditional network in [9], but the cost is high because several new devices are introduced. So collaborative changes among multiple network configurations demand powerful management of the network. Emerging software-defined network (SDN) [13] brings new method to realize dynamic network configuration. SDN decouples the control plane and the forwarding plane (data plane) and applies logic centralized control. The powerful network management and control ability of SDN make the realization of dynamic network configuration more flexible. The programmable nature of SDN can control flowtable of forwarding devices directly and avoid service interruptions and routing inflation. The centralized control of SDN makes it possible to have a global view of network. Therefore collaborative changes of multiple network configuration can be realized.

In this paper, double hopping communication (DHC) is proposed based on SDN architecture to enhance the ability to resist sniffer attack. DHC periodically changes the end information of both communication sides as well as the routing paths between them, thus realizing double hopping of end and route. In DHC, communication data is transmitted among multiple paths and data flow from multiple users will be mixed. It is difficult for attackers to obtain complete data from one communication in DHC and moreover it sets obstacles to avoid the attackers to correctly separate data of one single user among all the data they obtain. Therefore, overhead and difficulty for attackers to obtain and analyze communication data are dramatically increased due to the disability of attackers to conduct targeted sniffing. In addition, DHC is constructed based on SDN, which is transparent to the terminals and neither extra external software nor hardware is needed.

The rest of the paper is organized as follows. In Section 2 related works are discussed. Section 3 describes the basic principles of DHC. In Section 4 we describe the basic architecture and communication protocols of DHC. Section 5 presents the prototype deployment and simulation experiment and security of DHC are analyzed in Section 6. Section 7 concludes the paper.

2. Related Work

Hopping communication, based on dynamic and randomness of MTD technology, is one type of active network defense methods, aimed at breaking the hypothesis of static network configuration, and can improve network security via dynamic and randomness [11, 14]. Currently, researchers have proposed different hopping communication techniques. Atighetchi et al. [6] proposed a hopping approach based on fake address and port. Fake addresses and ports are used during data transmission to confuse attackers. Sifalakis et al. [15] proposed one network address hopping method (NAH) based on information hiding technique. Data flow is spread across multiple end-to-end connections by network address hopping during transmission. Thus point-to-point data transmission security could be improved. In [10] a random port-hopping (RPH) scheme was proposed to defend DDoS attacks by changing the communication ports. MT6D [16], proposed by Dunlop et al., taking the advantage of address space of IPv6 and robust IP hopping strategy, is achieved. Tunnel technique is used to encapsulate the packets. Source and destination IP addresses of the tunnel are changed repeatedly, making it difficult for attackers to sniff communication traffic. The approaches described above have their own advantages. However, in all of these methods, end is hopped, while routing path stays unchanged, which makes it possible for attackers to obtain complete communication data and therefore recover communication data. Moreover, in order to realize hopping communication, deploying software on terminal and adding hardware in the network are needed, which causes high cost.

In traditional network, quick cooperative hopping is difficult in distributed route management. However, the emerging software-defined network has brought new methods to hopping communication. Based on SDN, Kampanakis et al. [5] proposed three kinds of MTD methods, including reconnaissance protection, service version/OS hiding, and random host/route mutation. Attack cost, benefits, and potential attackers' countermeasures of these three methods are analyzed, respectively, in this work. These methods involve network scanning, DDoS, and worm, but DHC focuses on sniffer attack. In the SDN architecture, a flexible as well as transparent to terminal IP hopping method, called OF-RHM [7, 17], is proposed by Jafarian et al. It is true that the effectivity of sniffer attack is decreased by OF-RHM, but virtual IP should stay unchanged during one continuous communication, which enables attackers to obtain complete data of one communication from a switch. Jafarian et al. [18] proposed a technique in which hopping is implemented temporarily and spatially in order to interfere with attackers' views of the network. This hopping communication can defeat collaborative scanning attacks effectively. However, in our work, multiple network configurations are changed dynamically to enhance the dynamism of network for resisting sniffer attack. The work in [19] achieves fast IP hopping to resist scanning and worm propagation. The method discovers hazardous network ranges and addresses adaptively and evacuates network hosts from them quickly. MacFarland and Shue [20] provide a scalable moving target system to enable key security properties and maintain acceptable performance. The method distinguishes trustworthy and untrustworthy clients to provide access control for legacy clients.

There exist multiple paths between two nodes in network topology, which are used by researchers to improve communication security. An active random route mutation (RRM) method is proposed by Duan et al. [8, 21] and applied in SDN environment. Routes of multiple flows in the network are changed randomly and simultaneously. However, multiple uncrossed paths between source and destination are required, which is difficult to satisfy in common network topology. In addition, no end hopping is involved in RRM method, which enables attackers to recover communication data between hosts by sniffing multiple switches. Dolev and David [22] use multiple paths between datacenters to achieve secure communication. In order to ensure the privacy, an n-k secret sharing method is used to encrypt communication data. The

source creates n shares of its data, then sends them along multiple paths, and makes sure that no k or more shares pass the same router. Thus the method achieves theoretically secured channel to the public cloud. However, in our work, ends and route paths are changed frequently to increase the cost of attacks while obtaining and reconstructing communication data. Gillani et al. [23] migrate virtual routers among multiple paths to invalidate the network topology probe of attacks; therefore link DDoS attacks are resisted. Gkounis et al. [24] proposed a method based on SDN architecture to detect and mitigate Crossfire attack [25] by rerouting traffic via multiple paths. The two abovementioned works aim to resist link DDoS attacks, while our work, aimed at resisting sniffer attack, increases the cost of attackers through changes of ends and routing paths.

3. Basic Principles of DHC

In static configuration based network communication, when two hosts communicate on one connection, all the packets in communication contain information about this connection and the transmission path of the communication packets is static. These two facts provide convenience for attackers to sniff network communication. Attackers are able to obtain communication data easily from the target by sniffing network flow based on target end on transmission path. In DHC approach, both end and route are hopped based on SDN architecture. Dynamic and randomness are introduced in communication for two dimensions: end and route. For the data plane, random hopping end and route are configured by the controller in every hopping period after one connection is established. In the meantime, both end hopping and route hopping are achieved.

In DHC, ends in both communication sides hop dynamically. The data from multiple users will be mixed and end-to-end traffic is hidden in network background traffic. Frequent hopping of the end brings difficulty for attackers to select and sort the sniffed packets as well as recovering the initial data. Thus the difficulty of analyzing communication data is increased. Route hopping changes routing paths of the packets dynamically, spreading the communication traffic into multiple routing paths. In this way, overhead and difficulty of sniffing are increased since continuous communication data is difficult to obtain. To sum up, double hopping of both end and route limits the communication data that attackers can obtain and set obstacles for attackers to analyze the data.

4. Basic Architecture of DHC

When conducting hopping communication in DHC, end and routing path that are about to hop are selected first. Then flowtables are updated according to hopping protocol. Thus end hopping space and route hopping space as well as hopping communication protocol should be taken into consideration to realize DHC.

4.1. End Hopping Space. End consists of IP address of the host and port in communication. It is an essential element of communication between two hosts in network and it uniquely defines one communication side in network. One connection in network communication contains IP addresses and ports of both source and destination hosts. Therefore, $EI = (\mathrm{IP}_{\mathrm{src}}, P_{\mathrm{src}}, \mathrm{IP}_{\mathrm{dst}}, P_{\mathrm{dst}})$ is defined to represent the end of one connection. End of packets mentioned through the paper refers to this definition. In DHC, end hopping space S_{EH} consists of hopping IP addresses and hopping ports. Given IP address pool $\mathrm{Addr} = \{\mathrm{IP}_1, \mathrm{IP}_2, \ldots, \mathrm{IP}_m\}$ and hopping port pool $\mathrm{Port} = \{P_1, P_2, \ldots, P_n\}$, end hopping space can be represented by

$$S_{\mathrm{EH}} = \{(\mathrm{IP}_{\mathrm{src}}, P_{\mathrm{src}}, \mathrm{IP}_{\mathrm{dst}}, P_{\mathrm{dst}}) \mid \mathrm{IP}_{\mathrm{src}}, \mathrm{IP}_{\mathrm{dst}} \in \mathrm{Addr}, \mathrm{IP}_{\mathrm{src}} \neq \mathrm{IP}_{\mathrm{dst}}, P_{\mathrm{src}}, P_{\mathrm{dst}} \in \mathrm{Port}\}. \quad (1)$$

Unoccupied hopping ends are randomly selected in S_{EH} to replace the real ends in communication when the ends need hopping.

4.2. Route Hopping Space. One routing path between source and destination hosts is a sequence that consists of forwarding nodes (i.e., OF switch). Define $Path = \langle node_{\mathrm{src}}, node_1, node_2, \ldots, node_{\mathrm{dst}} \rangle$, where $node_{\mathrm{src}}$ connects with source host and is called source forwarding node (source switch); $node_{\mathrm{dst}}$ connects with destination host and is called destination forwarding node (destination switch). Under SDN architecture, controller has the global network view. Therefore, all paths connecting source and destination hosts that satisfy certain conditions can be calculated, constituting the route hopping space.

Suppose the source host H_1 communicates with destination host H_2; the corresponding route hopping space $S_{\mathrm{RH}}^{H_1 \to H_2}$ will be calculated as follows:

(1) Calculate all acyclic paths between H_1 and H_2 that are not longer than the maximum path length L according to the topology of network and constitute the path set $PathSet^{H_1 \to H_2}$.

(2) For $Path_i, Path_j \in PathSet^{H_1 \to H_2}$, if $Nodes(Path_i) \subset Nodes(Path_j)$ holds, delete $Path_j$ from path set $PathSet^{H_1 \to H_2}$, where $Nodes(Path_i)$ represents the set of nodes that path $Path_i$ passes. The reason for deleting $Path_j$ is that no node in $Path_i$ can be avoided when packets pass along $Path_j$, which leads to a longer path.

The route hopping space $S_{\mathrm{RH}}^{H_1 \to H_2}$ is obtained from the steps above. If $|S_{\mathrm{RH}}^{H_1 \to H_2}| > 1$ holds, the paths in $S_{\mathrm{RH}}^{H_1 \to H_2}$ satisfy the following: $\forall Path_i, Path_j \in S_{\mathrm{RH}}^{H_1 \to H_2}$, $Path_i \neq Path_j$, there exists $node \in Nodes(Path_i)$ and $node \notin Nodes(Path_j)$, which means that $Path_j$ does not pass at least one node in $Path_i$.

In order to guarantee the unpredictability of the hopping path, randomness in hopping path selection is essential. One simple method is random path selection which randomly selects one path in $S_{\mathrm{RH}}^{H_1 \to H_2}$ at the beginning of each period and takes it as the hopping path during the period. The probability of selection for each path in $S_{\mathrm{RH}}^{H_1 \to H_2}$ is identical.

Input: $(Path_1, \ldots, Path_u)$, $(w_1, \ldots, w_u)$, *RandNum*
Output: *Path*
WeightedRandomPathSelect$((Path_1, \ldots, Path_u), (w_1, \ldots, w_u), RandNum)$
(1) sum = 0
(2) **for** i in $(1, 2, \ldots, u)$
(3) new_sum $\leftarrow$ sum + w_i
(4) **if** sum < *RandNum* $\leq$ new_sum
(5) **return** $Path_i$
(6) **else** sum = new_sum

Algorithm 1: Weighted random path selection algorithm.

However, traffic may be forwarded unbalanced by the nodes, which means possibility of large amount of traffic forwarded by one single node exists. In this case, if attackers sniff on this specific node, large amount of communication data will be obtained easily. The reason is that paths in $S_{\mathrm{RH}}^{H_1 \to H_2}$ intersect. Fortunately, this threat can be eliminated in DHC by using weighted random path selection.

For a node, we define $C^{H_1 \to H_2}(node)$ as the number of paths in route hopping space $S_{\mathrm{RH}}^{H_1 \to H_2}$ that pass through *node*. For a node set *set*, we define $C^{H_1 \to H_2}(set) = \{C^{H_1 \to H_2}(node) \mid node \in set\}$. Suppose that, for one connection between hosts H_1 and H_2, there is $Path_k \in S_{\mathrm{RH}}^{H_1 \to H_2}$. $d(Path_k)$ donates the node set that contains the nodes left after common nodes (e.g., source forwarding node and destination forwarding node) through which all paths in $S_{\mathrm{RH}}^{H_1 \to H_2}$ pass are deleted. The weight of $Path_k$ is defined

$$Weight\left(Path_k\right) = \frac{1/\operatorname{Max}\left(C^{H_1 \to H_2}\left(d\left(Path_k\right)\right)\right)}{\sum_{Path_i \in S_{\mathrm{RH}}^{H_1 \to H_2}} \left(1/\operatorname{Max}\left(C^{H_1 \to H_2}\left(d\left(Path_i\right)\right)\right)\right)}, \quad (2)$$

where the function Max gets the maximum value in $C^{H_1 \to H_2}(set)$. By using the weighting function above, lower weight is assigned to paths with nodes that more paths cross. Therefore, chances for overmuch traffic passes through one single node (except common nodes for all paths) in network due to intersection are eliminated.

Weighted random path selection algorithm is shown in Algorithm 1. The probability of one path to be chosen is set as the weight for the path. The inputs of the algorithm include paths $(Path_1, \ldots, Path_u)$ in route hopping space $S_{\mathrm{RH}}^{H_1 \to H_2}$, corresponding weights $(w_1, \ldots, w_u)$, and a random number $RandNum \in [0, 1]$. In the algorithm, weights are accumulated for each path in steps 2 to 6. The path corresponding to the weight is returned when the sum of accumulated weights is bigger than or equal to the random number *RandNum*.

4.3. DHC Protocol. In DHC, for each period T_{hop}, one hopping end *hEI* and one path from source to destination *Path* are randomly chosen. New flow entries are generated by the controller and installed in OF switches. End of packets from source host is modified to *hEI* and these packets are transmitted to destination host along *Path*. Then double hopping of end and route with period T_{hop} as granularity is realized.

4.3.1. Double Hopping. The basic protocol of DHC is illustrated in Figure 1. It is a network with SDN architecture, in which host H_1 communicates with H_2. Denote end hopping space as S_{EH} and route hopping space of the communication as $S_{\mathrm{RH}}^{H_1 \to H_2}$. Firstly, initial end $rEI = (\mathrm{IP}_1, P_1, \mathrm{IP}_2, P_2)$ is generated by H_1 according to the real IP address and port of two communication sides; then the address of the communication is determined.

Detailed steps of double hopping are as follows:

(1) The first packet containing *rEI* is sent to the network by H_1. OF switch S_1 receives the packet and encapsulates it as a packet-in message. Then the packet-in message is sent to the controller.

(2) The packet-in message is deencapsulated by the controller and *rEI* is extracted. Then hopping end $hEI_1 = (\mathrm{IP}_1', P_1', \mathrm{IP}_2', P_2')$ is selected randomly in S_{EH}. Route hopping space $S_{\mathrm{RH}}^{H_1 \to H_2}$ is calculated by the controller and $Path_1 = (S_1, S_2, S_5)$ is chosen using weighted random path selection algorithm. With the knowledge of hEI_1 and $Path_1$, controller generates flow entries encapsulated as modify-state messages and sends them to OF switches S_1, S_2, and S_5. Corresponding modification and routing of the packets are conducted.

(3) Ends $(\mathrm{IP}_1, P_1, \mathrm{IP}_2, P_2)$ in the packets are modified to $(\mathrm{IP}_1', P_1', \mathrm{IP}_2', P_2')$ by source switch S_1 and the modified packets are forwarded to OF switch S_2 then to destination switch S_1.

(4) Ends $(\mathrm{IP}_1', P_1', \mathrm{IP}_2', P_2')$ in the packets are recovered to the *rEI* and forwarded to host H_2 by destination switch S_5. Then H_2 receives the packets from H_1.

In this communication, the hopping end is recalculated by the controller for a hopping period T_{hop} and is represented as $hEI_2 = (\mathrm{IP}_1'', P_1'', \mathrm{IP}_2'', P_2'')$ as shown in Figure 1. A new path, denoted as $Path_2 = (S_1, S_3, S_4, S_5)$, is selected in $S_{\mathrm{RH}}^{H_1 \to H_2}$ using weighted random path selection algorithm. Then the flow entries in OF switches are updated. Source switch S_1 modifies the end in the packets sent from H_1 to H_2 as hEI_2 and

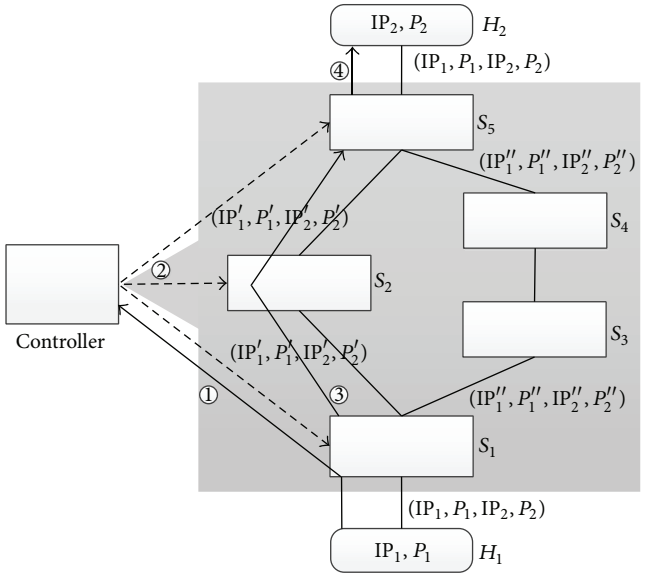

Figure 1: An example of double hopping communication.

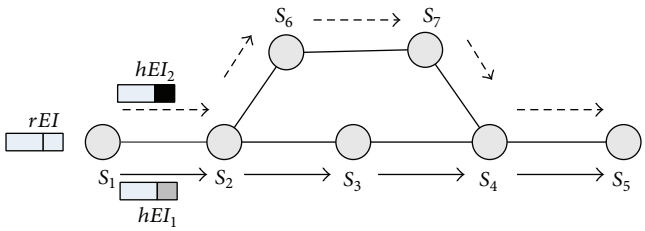

Figure 2: An example of flow entries update.

forwards the modified packets along $Path_2$. In destination switch S_5, the end of these packets is recovered to the real end (IP_1, P_1, IP_2, P_2).

The procedure described above does not modify the real end on both hosts. Instead it modifies the end and routing path of the communication packets dynamically in network transmission. The source and destination hosts can achieve hopping communication transparently in network without interrupting the ongoing communication. Once the packets of communication between H_1 and H_2 enter the network, end of the packets and routing path are hopped with time. For each hopping period T_{hop}, the hopping end and route will be reconfigured by the controller. The communication will be considered finished when the controller detects the fact that the flow entries are not hit in a hopping period via flow-removed messages sent by switches. Thus flow entries will not be updated.

4.3.2. Flow Entries Update. Flow entries in OF switches need to be updated when end and route are hopped in DHC. Moreover, it should be guaranteed that the flow entries update is consistent and no packet is lost. Suppose that hopping communication is conducted in the network topology as shown in Figure 2. Assume that the end is being hopped by switch S_1 currently, end changes from rEI to hEI_1, and the packets are being transmitted along path $(S_1, S_2, S_3, S_4, S_5)$. At this circumstance, to hop the end of the packets from rEI to hEI_2 and to hop the routing path from $(S_1, S_2, S_3, S_4, S_5)$ to $(S_1, S_2, S_6, S_7, S_4, S_5)$, the steps of updating flow entries are as follows:

(1) Controller sends modify-state messages to install new flow entries in switches S_2, S_6, S_7, S_4, S_5 for forwarding the packets with end hEI_2. At this time, the new flow entries will not be hit by packets, because there are no packets in the network that contain the end hEI_2.

(2) Controller sends modify-state messages to modify the flow entry in switch S_1; thus the end of packets is converted from rEI to hEI_2.

(3) Controller sends modify-state messages to delete the old flow entries in switches S_2, S_3, S_4, S_5 after the maximum transmission delay of path $(S_1, S_2, S_3, S_4, S_5)$ is reached.

The method to update the flow entries described above can guarantee that the traffic is routed by the old flow entries during update, avoiding packets loss. In addition, traffic is routed by the updated flow entries after update, maintaining per-packet consistency.

5. Prototype Deployment and Simulation Experiment

5.1. Prototype Deployment. To verify the performance and security of DHC, DhcFlower, a prototype based on SDN controller is implemented. As shown in Figure 3, DhcFlower runs on the top of SDN controller which manages switches through OpenFlow.

In the prototype deployment of DHC, TopologyDiscovery reports the changes of network topology and updates view of network. FlowMonitor monitors the flow state of network to find initiation and termination of connections. Based on the view and flow state of network, DhcFlower chooses the ends and routing paths to convert network configurations.

Detailed structure of DhcFlower is shown in Figure 4. TopologyDiscovery updates topology database TopologyInfo with the changes of network topology. Using the network topology information, hopping path calculator calculates multiple paths of each pair of nodes and stores hopping path information in the hopping path pool. Hopping ends are stored in Hopping end pool. With hopping end pool and hopping path pool, double hopping engine, as the core module, chooses the hopping end and path based on flow state information. Afterwards, strategies of hopping are generated. Flow updater generates flow entries based on hopping strategies and updates the flowtables in a specific order.

5.2. Simulation Experiment. To evaluate DHC, we have operated our implement prototype over the Mininet [26]. OpenFlow 1.0 [27] is applied and POX [28] is used as controller. A class B address block is chosen as hopping IP address pool and hopping port pool denoted as $\{0, 1, \ldots, 65535\}$. Network

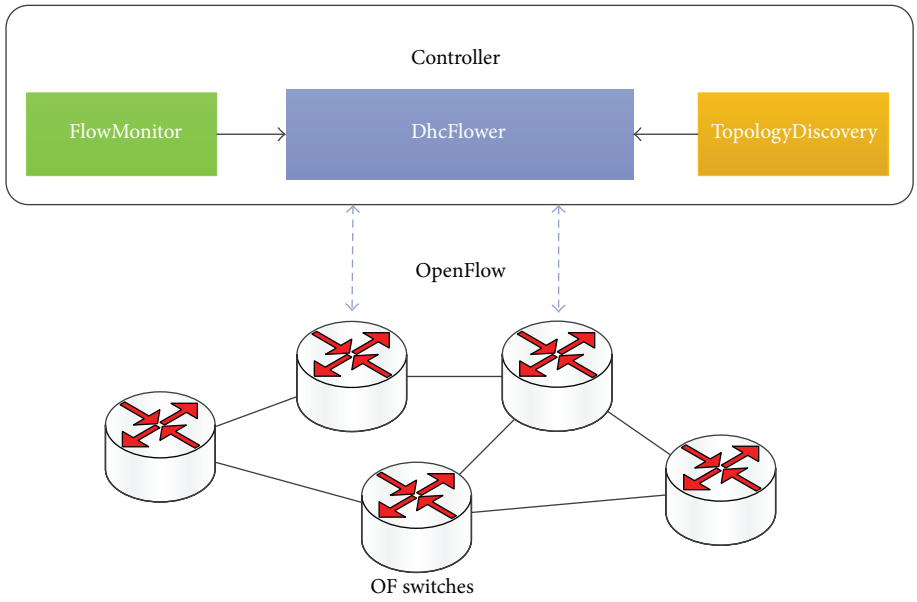

Figure 3: DHC prototype deployment.

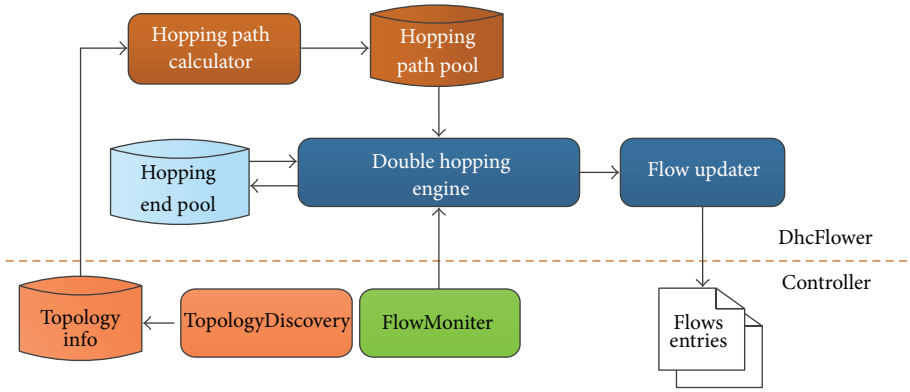

Figure 4: Makeup of DhcFlower.

topology proposed by [29] is applied, which has 16 nodes (forwarding nodes) as illustrated in Figure 5. The maximum path length L is set to 32.

5.2.1. Validation of the Effectiveness of End Hopping. UDP packets from terminal on node 1 are sent to terminal on node 16 for 500 s. Packets are sniffed on the forwarding nodes and the number of ends received on each node is counted. The sniffing results in DHC and traditional network are shown in Figure 6.

As demonstrated in Figure 6, on some forwarding nodes in traditional network, such as nodes 4, 7, 8, and 12, only one end is able to be sniffed. However, in DHC, apart from source and destination forwarding nodes, multiple ends can be sniffed on other forwarding nodes. Due to the invariant of packets' end in traditional networks, end that is sniffed stays unchangeable, which brings convenience for attackers. Attackers can launch a targeted sniffer to any connection and obtain the complete communication data of the connection. In DHC, end changes randomly and periodically. The ends sniffed on forwarding nodes between source and destination hosts are various. It is difficult for attackers to determine the ends from the same connection, increasing the difficulty in reconstructing the communication data. Moreover, the more frequently ends hop, the more ends will be sniffed on forwarding nodes. It can be seen in Figure 6 that more ends are sniffed when $T_{\text{hop}} = 5\,\text{s}$ compared with $T_{\text{hop}} = 10\,\text{s}$. In addition, fewer ends can be sniffed on forwarding node 9 than other nodes as can be seen in the figure. The reason is that fewer paths pass through forwarding node 9 than other nodes; thus the probability of being hit by weighted random selection is lower.

5.2.2. Validation of the Effectiveness of Route Hopping. In the experiment, 10^6 packets are transmitted from node 5 to node 6 with the speed of 10^4 packets per second. The hopping period T_{hop} is set to 5 s. Packets are sniffed on the forwarding nodes and the number of packets sniffed

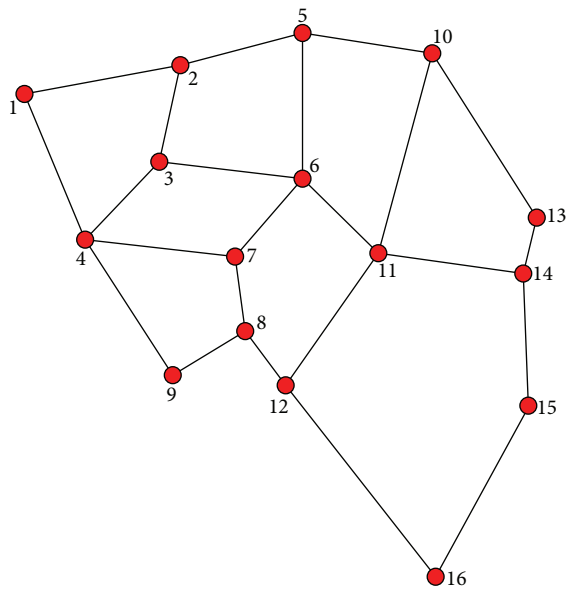

Figure 5: Network topology applied in the experiment.

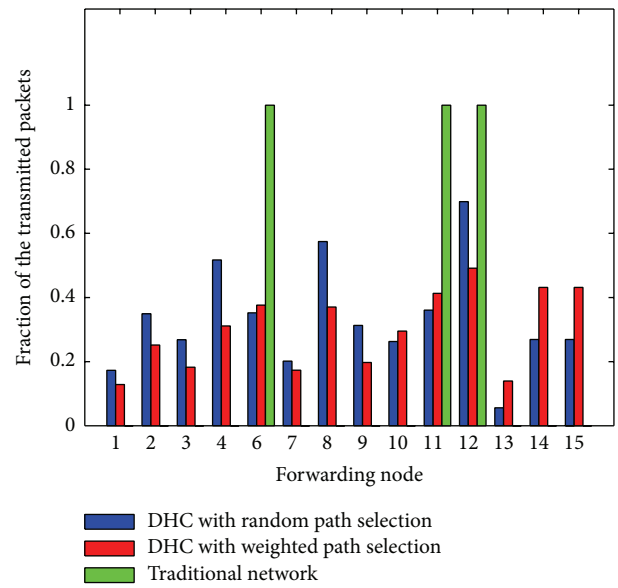

Figure 7: Percentage of packets sniffed from single flow.

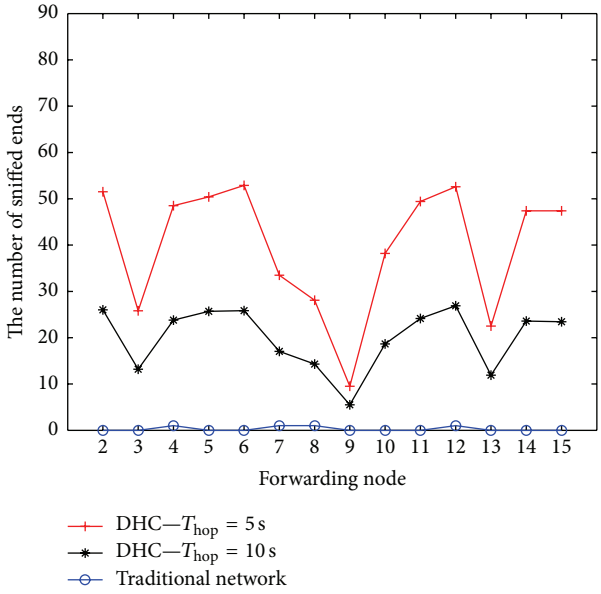

Figure 6: Number of ends sniffed from single flow.

is counted. In DHC network, random path selection and weighted random path selection are applied to conduct hopping communication. Sniffing results are compared with traditional network communication, as shown in Figure 7.

In Figure 7, the vertical coordinate stands for the fraction of all the packets transmitted from node 5 to node 6. As we can see, in traditional network, complete communication data from source host to destination host can be sniffed on some nodes (e.g., nodes 6, 11, and 12), which means that attackers can sniff complete data on any of the nodes and further data analysis is possible. Since shortest-path routing is applied in traditional network and the path stays unchanged during communication, the complete communication data can be obtained on any node that the shortest path goes through. In DHC, packets of a connection are distributed to several paths by route hopping. It is difficult for attackers to sniff complete data on single forwarding node. Possibility for sniffing large amount of data on a certain nodes exists if random path selection is applied. As shown in Figure 7, more than 50% of the data can be sniffed on forwarding nodes 4, 8, and 12. Applying weighed random path selection can avoid excessive traffic passing through certain nodes. The reason is that lower weight is assigned to paths with nodes that more paths cross.

5.2.3. Validation of Effectiveness of Antisniffer Attack. In the experiment, 100 MB data had been transmitted from node 1 to node 16 for 500 s. The hopping period T_{hop} is set to 5 s. Data is sniffed on node sets $A1 = \{8\}$, $A2 = \{8, 9\}$, $A3 = \{8, 9, 10\}$, and $A4 = \{8, 9, 10, 11\}$, respectively. The shortest path from node 1 to node 16 is $1 \rightarrow 4 \rightarrow 7 \rightarrow 8 \rightarrow 12 \rightarrow 16$. The percentage of data sniffed on node sets $A1$, $A2$, $A3$, and $A4$ is presented in Figure 8.

As illustrated in Figure 8, complete communication data can be sniffed on all sniffed node sets, $A1$, $A2$, $A3$, and $A4$, in traditional network since they all contain node 8 on the shortest path, on which complete data can be sniffed. However, in DHC, complete data cannot be obtained from node sets $A1$, $A2$, and $A3$ since route hopping is applied. The percentage of data sniffed on $A1$ and $A2$ is the same because traffic passes through $A2$ and also passes through $A1$. Only $A4$ can sniff the complete communication data in DHC. However ends of the data are diverse because of end hopping. We consider that packets with the same end are static data that attackers can obtain. The static data that attackers can obtain in hopping communication is far less than that in traditional network.

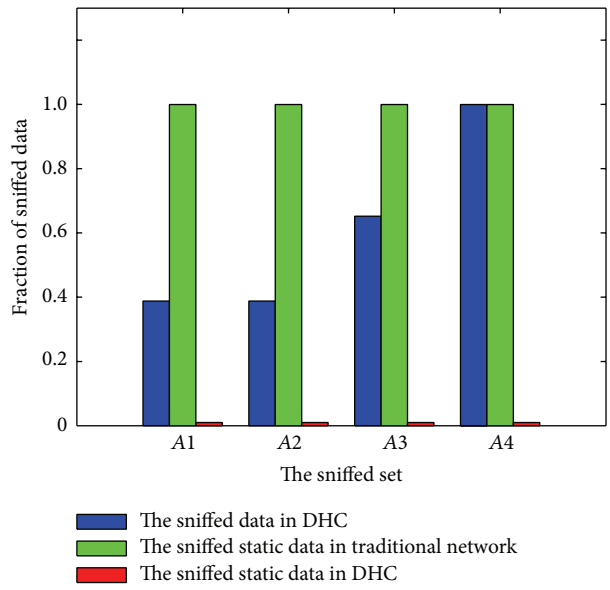

Figure 8: Percentage of data that can be sniffed by attackers.

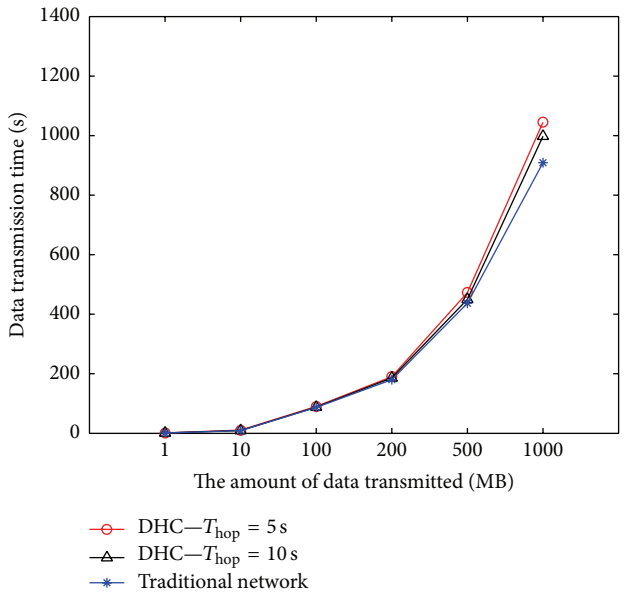

Figure 9: Performance of forwarding in DHC.

5.2.4. Performance of DHC. In the experiment, bandwidth of all connections in network topology is set to 10 Mb/s. Data is transmitted from terminal on node 1 to terminal on node 16 using File Transfer Protocol (FTP). Time for data transmission in both DHC and traditional network is recorded. Results are shown in Figure 9.

As can be seen in Figure 9, time consumption of data transmission in DHC increased in comparison with traditional network. The reason is that multiple paths from source to destination are selected, including longer paths. On the contrary, the data is routed by the shortest path in traditional network. Therefore, transmission time in DHC is longer than that in traditional network. But the increase is less than 7% when T_{hop} = 5 s in the experiment. Routing path hopping of a connection results in a small amount of disordered packets at receiving end when new period starts. Then retransmission is caused. Therefore the more frequently the entries update flow, the more likely the retransmission happens. We can also see from Figure 9 that longer time will be consumed to transmit data when T_{hop} = 5 s compared with T_{hop} = 10 s.

6. Analysis

In DHC, each hopping connection needs to occupy hopping ends in every period. In Section 6.1, the number of hopping connections that can be supported in DHC network, that is, hopping network capacity, is analyzed. DHC brings difficulty for attackers to obtain complete data and to reconstruct data. Therefore, communication security is improved. The obtaining and reconstruction of communication data are discussed in Sections 6.2 and 6.3. The unpredictability and the cost of DHC are analyzed in Sections 6.4 and 6.5, respectively.

6.1. Capacity of Hopping Network. Suppose the sizes of hopping IP address pool and port pool are |Addr| and |Port|, respectively. The number of all the ends $(IP_{src}, P_{src}, IP_{dst}, P_{dst})$ is $|Addr|^2 \times |Port|^2$, and the number of the ends is $|Addr| \times |Port|^2$ when $IP_{src} = IP_{dst}$. According to the definition of end, valid ends require $IP_{src} \neq IP_{dst}$, so the size of valid end hopping space S_{EH} can be calculated by

$$|S_{EH}| = |Addr|^2 \times |Port|^2 - |Addr| \times |Port|^2. \quad (3)$$

In DHC, end hopping is performed in both directions of one connection, which means that, at any moment, one connection needs two ends. Assuming t hopping connections exist simultaneously in network, $2t$ ends will be needed, so $|S_{EH}| - 2t$ ends are left. To ensure high randomness in hopping end selection, enough unoccupied hopping ends in S_{EH} are necessary. Suppose the maximum occupancy rate in end hopping space S_{EH} is α; that is, there are at least $(1-\alpha)|S_{EH}|$ ends unoccupied. Then inequality (4) holds:

$$\begin{aligned} (1-\alpha)|S_{EH}| &\leq |S_{EH}| - 2t \\ t &\leq \frac{1}{2}\alpha|S_{EH}|. \end{aligned} \quad (4)$$

Therefore, the maximum number of hopping connections allowed in DHC is $(1/2)\alpha|S_{EH}|$; that is, the capacity of hopping network is $(1/2)\alpha|S_{EH}|$.

Combining (3) and inequality (4), the following inequality can be obtained:

$$t \leq \frac{1}{2}\alpha\left(|Addr|^2 \times |Port|^2 - |Addr| \times |Port|^2\right). \quad (5)$$

Assume $|Port| = 2^{16}$, $|Addr| = 2^{16}$ (hopping IP address pool is a class B address block), and $\alpha = 0.8$; DHC can support 7.37×10^{18} connections hopping simultaneously.

6.2. Analysis of Complete Communication Data Obtaining by Attackers. We hypothesize that attackers can sniff part of the

forwarding nodes in network randomly. Suppose network topology $G = \langle V, E \rangle$ is an undirected connected graph, where V is a set of forwarding nodes and E is a set of links. V contains m forwarding nodes and attackers can randomly sniff n of them simultaneously ($n \leq m$). Sniffed node set consisting of these sniffed forwarding nodes is denoted as V^n_{sniff}. $V^n_{\text{sniff}} \subseteq V$ and $|V^n_{\text{sniff}}| = n$.

Source host $host_{\text{src}}$ communicates with destination host $host_{\text{dst}}$. Source and destination forwarding nodes are denoted as $node_{\text{src}}$ and $node_{\text{dst}}$, respectively. Assume there are s nodes on the shortest path between $host_{\text{src}}$ and $host_{\text{dst}}$ ($1 \leq s \leq m$), which constitute node set U^s. In traditional network, if $V^n_{\text{listen}} \cap U^s \neq \varnothing$, complete communication data between $host_{\text{src}}$ and $host_{\text{dst}}$ can be obtained by attackers. If $V^n_{\text{listen}} \cap U^s = \varnothing$, no communication data can be sniffed. The probability of attackers obtaining complete communication data in traditional network can be calculated by (6), where C^n_m is number of all V^n_{sniff} and C^n_{m-s} is the number of V^n_{sniff} when $V^n_{\text{listen}} \cap U^s = \varnothing$. So $C^n_m - C^n_{m-s}$ represents the number of V^n_{sniff} when $V^n_{\text{listen}} \cap U^s \neq \varnothing$:

$$P_{\text{traditional}} = \frac{C^n_m - C^n_{m-s}}{C^n_m}. \tag{6}$$

In DHC, attackers can sniff complete data between $host_{\text{src}}$ and $host_{\text{dst}}$ if $node_{\text{src}} \in V^n_{\text{sniff}}$ or $node_{\text{dst}} \in V^n_{\text{sniff}}$. The number of such V^n_{sniff} is $C^1_2 C^{n-1}_{m-2} + C^2_2 C^{n-2}_{m-2}$. In other cases, if $node_{\text{src}} \notin V^n_{\text{listen}}$ and $node_{\text{dst}} \notin V^n_{\text{listen}}$, to sniff complete data, one vertex cut-set V_{cut} should be contained in V^n_{sniff}, and $node_{\text{src}}$ and $node_{\text{dst}}$ should be cut by V_{cut} into different connected subgraphs; that is, $V^n_{\text{sniff}} \supseteq V_{\text{cut}}$ exists, where G is cut by V_{cut} into k connected subgraphs $G_1, G_2, \ldots, G_k$, and $node_{\text{src}} \in G_i$ and $node_{\text{dst}} \in G_j$, $1 \leq i, j \leq k$, and $i \neq j$, hold. Suppose there exists $Q^n_{\text{src,dst}}$ sniffed node set V^n_{sniff}, where V^n_{sniff} contains such V_{cut} in this case. Then the probability of attackers obtaining complete data between $host_{\text{src}}$ and $host_{\text{dst}}$ can be calculated by

$$P_{\text{hop}} = \frac{C^1_2 C^{n-1}_{m-2} + C^2_2 C^{n-2}_{m-2} + Q^n_{\text{src,dst}}}{C^n_m}. \tag{7}$$

Proposition 1. *The probability of attackers obtaining complete data in traditional network on one communication is not less than that in DHC; that is,* $P_{\text{traditional}} \geq P_{\text{hop}}$.

The proof process of this proposition is shown in the Appendix. In the network topology shown in Figure 5, suppose a host on node 1 communicates with a host on node 16. The shortest path from node 1 to node 16 contains 6 nodes. Attackers can sniff n nodes randomly ($1 \leq n \leq 16$). Probabilities of attackers obtaining complete data in traditional network and DHC network are shown in Figure 10.

As can be seen from Figure 10, probability of attackers obtaining complete data increases when number of sniffed nodes increases, both in traditional and DHC network. But $P_{\text{hop}} \leq P_{\text{traditional}}$ always holds. Probability of attackers obtaining complete data is 1 in both traditional and DHC network when the number of sniffed nodes is more than 10. Although probability of attackers sniffing complete data increases in DHC network when large number of forwarding nodes are sniffed, attackers obtain more irrelevant data. Since end hops constantly during a communication, attackers cannot pick out the traffic that belongs to the target from the sniffed data easily, which increases the difficulty for attackers to reconstruct and recover communication data.

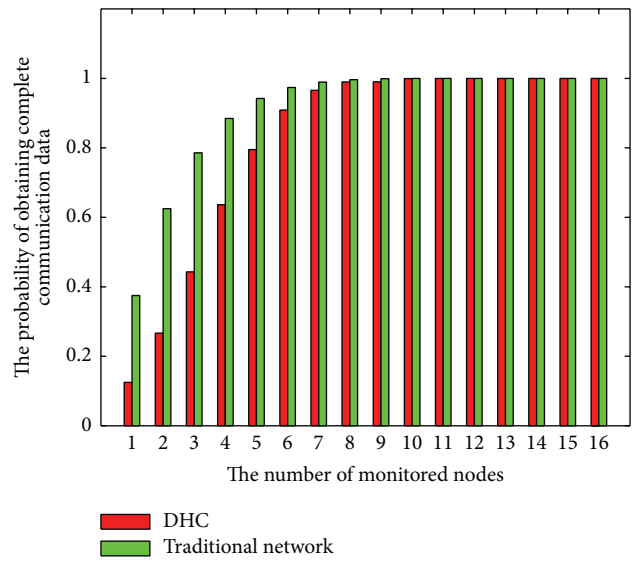

Figure 10: Probability of obtaining complete data.

6.3. Analysis of Communication Data Reconstruction for Attackers. Reconstruction of communication data requires complete data in this communication. Assume attackers can sniff complete data in communication between source and destination hosts in this section. In traditional network, attackers can deduce the positions of both communication sides and upper layer protocol according to IP and port of the sniffed packets. Useless packets can be eliminated based on the end and the target communication data can be obtained. However, in DHC network no real end from source and destination hosts can be sniffed by attackers if source and host forwarding nodes are not sniffed. Data in communication is distributed to various flows that attackers are not able to distinguish. Suppose that there are f_{sniff} flows in the sniffed data, among which f_{real} flows contain the data of target connections ($f_{\text{real}} \leq f_{\text{sniff}}$) and different ends are applied in different connections. There are $C^h_{f_{\text{sniff}}}$ combinations since attackers randomly choose h flows from f_{sniff} flows. Attackers can reconstruct communication data properly with only one combination; that is, $C^{f_{\text{real}}}_{f_{\text{real}}} = 1$. Given that attackers select several flows randomly for a single time to reconstruct communication data, probability of reconstructing data properly can be calculated with

$$P_{\text{once}} = \frac{C^{f_{\text{real}}}_{f_{\text{real}}}}{C^1_{f_{\text{sniff}}} + C^2_{f_{\text{sniff}}} + \cdots + C^{f_{\text{sniff}}}_{f_{\text{sniff}}}} = \frac{1}{2^{f_{\text{sniff}}} - 1}. \tag{8}$$

As shown in (8), probability of attackers reconstructing data successfully with a single time decreases exponentially with

Table 1: Comparison of packet transmission time between traditional network and DHC network.

Approach	Average cost of packet transmission time	Period of flow update	Routing path
Traditional	$t \times l_s$	Infinite	The shortest path from source to destination
DHC	$t \times l_a$	T_{hop}	Multiple paths from source to destination

the increase of number of flows sniffed. The more data sniffed, the more difficulties for successful data reconstruction. Since attackers cannot determine the timing of target communication easily due to end hopping, longer sniffing time is needed to obtain complete communication data. Therefore large amount of irrelevant data is obtained, increasing the difficulty for data reconstruction. Given $f_{\text{sniff}} = 100$ and $f_{\text{real}} = 10$, the probability of attackers reconstructing data correctly by selecting several flows randomly for one time would be 7.89×10^{-31}.

6.4. Analysis of Unpredictability. Since the end and route hop randomly in DHC (detailed information is illustrated in Section 4.3), the end and route used in next period can not be predicted precisely. Under the condition of exposing DHC protocol, end hopping space, and route hopping space, DHC can still increase the cost of sniffer attackers and resist sniffer attacks. Suppose that an attacker with all the information above sniffs the DHC network for a target communication, then she will face the following difficulties in launching sniffer attack. Firstly, even though DHC protocol is transparent to the attacker, a targeted sniffer attack can not be launched thanks to the randomness of end and route hopping. Secondly, it is hard for the attacker to get complete communication data during sniffing due to periodical hopping of route. Thirdly, the attacker will get a large number of ends because of frequent end hopping, which prevents the attacker from extracting the right packets belonging to the target communication when she/he attempts to recover communication data. So the unpredictability of DHC guarantees that it can resist sniffer attack under the condition of exposing DHC protocol and network information.

6.5. Analysis of Cost. Under traditional routing schemes, the packets are routed along the shortest path. However, in DHC network, packets may be routed along longer paths due to dynamic changing of the route. Therefore the cost of packet transmission time is higher in DHC. Let l_s denote length (the length of a routing path is estimated by hops) of the shortest path between source and destination, l_a the average length of paths in route hopping space ($l_s \leq l_a$), and T_{hop} the hopping period, then the cost of packet transmission time is shown in Table 1. Moreover, random selection of routing is periodically conducted by routing path hop of a communication, which results in a small number of disordered packets at receiving end when a new period starts, leaving no obstacles to normal communication.

Ends and routing paths will be selected in DHC when flow entries are generated, which is more complicated than that in traditional network. Therefore time cost of generating flow entries is higher in DHC. Since average path is longer in DHC, more flow entries are installed for one communication compared with traditional network. Thus the time cost for flow entries setup is higher in DHC as well. In Figure 11, the average time cost for installing flow entries between different node pairs in topology (shown in Figure 5) of DHC and traditional network is compared. As illustrated in Figure 11, the average time for flow entries generation and setup in DHC is longer than that in traditional network.

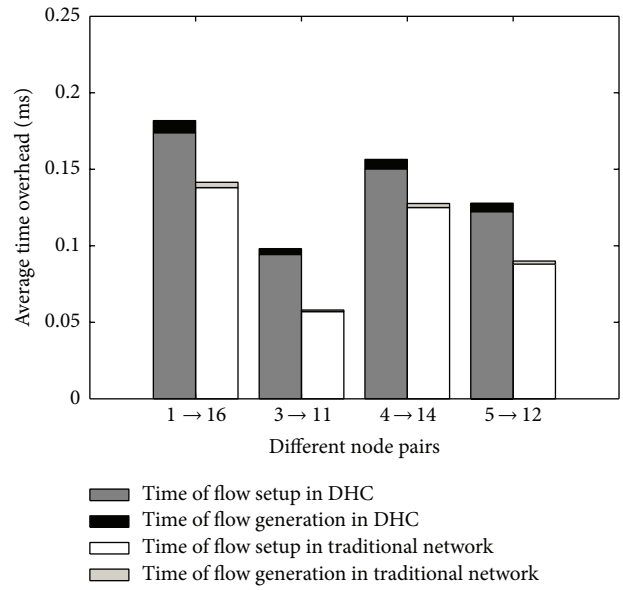

Figure 11: Comparison average time cost of flow entries installation in DHC and traditional network.

In the network without DHC, flow entries are installed only once at the beginning of communication, while in DHC flow entries of data plane are updated periodically and hopping ends and paths have to be allocated for any connection of two communication sides, which brings more loads for the controller. In experiment topology, 50 pairs of source and destination hosts are chosen randomly and communication between any pairs is stared. The CPU utilization of DHC and traditional network is compared in Figure 12. If controller does not run DHC, the load is low because the flow entry is not periodically updated. Therefore, the CPU utilization is under 10% as shown in Figure 12. If a controller runs DHC, the load increases due to periodical updating of flow entries. It can be found in the figure that CPU utilization is much higher when controller runs DHC. When $T_{\text{hop}} = 5$ s, the CPU utilization is between 20% and 40% and when $T_{\text{hop}} = 10$ s the CPU utilization is between 10% and 30%. The shorter hopping period enables more controller operations. So when $T_{\text{hop}} = 5$ s, CPU utilization of a controller is higher than when $T_{\text{hop}} = 10$ s. Controller will be the bottleneck when DHC

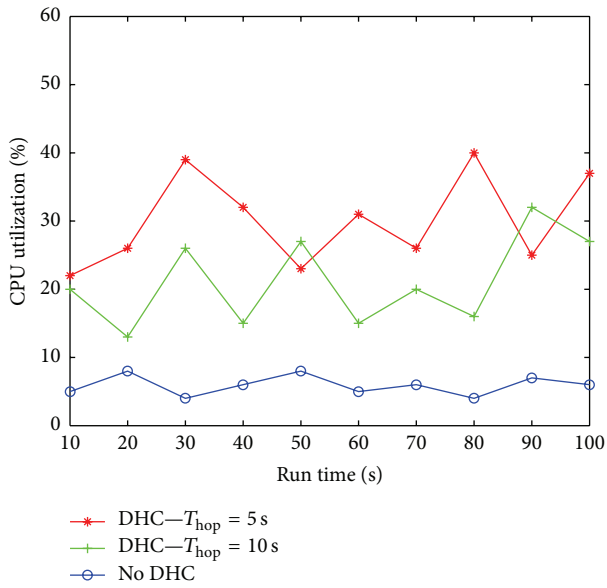

Figure 12: CPU utilization of controller.

is used in large scale network. Fortunately, distributed SDN controller [30] is a solution to the problem.

In traditional network, flows are matched only by destination addresses. So the length of routing tables is an order of $O(m)$ given the network of m nodes. However, flows are matched by ends (including source/destination address and ports) in DHC, meaning that two flows must be specified for every connection (TCP or UDP) between two communication sides. Let λ denote the average speed of connection establishment and let w denote the lasting time of each connection; then the mean length of flowtables is an order of $O(m\lambda w)$ [7]. Moreover, to avoid packets loss, DHC requires both old and new flow entries in flowtable simultaneously for a brief period of time, during which the cost of flowtable space increases. Therefore the cost of flowtable space is higher in DHC.

7. Conclusion

The centralized control and programmability of SDN make hopping communication easier to realize and deploy. In this paper, end hopping and route hopping are combined and double hopping communication based on SDN is proposed. End is changed dynamically in DHC so that the data from multiple users is mixed and communication traffic can be hidden in background traffic. So traffic cannot be distinguished easily and the difficulty for attackers to reconstruct and recover data increases. In addition, the data is transmitted along multiple paths by changing routing path dynamically. The difficulty for attackers to obtain complete communication data is increased. Results show that the approach proposed in this paper effectively enables antisniffer. Moreover, DHC is realized completely based on software and also transparent to terminals. Controller bottleneck usually occurs in large scale network of DHC. In the future work, a distributed controller model will be applied to deal with the problem and feasible communication solution of DHC will be tested in real network.

Appendix

Suppose there are m nodes in network topology G. Attacker can sniff n nodes and the sniffed nodes constitute a sniffed node set V^n_{sniff} ($|V^n_{\text{sniff}}| = n,\ n \le m$). Given the route hopping space $S^{H_1 \to H_2}_{\text{RH}}$, there are s nodes in the shortest path between source host H_1 and destination host H_2 ($s \le m$). V_{cut} is a vertex cut-set by which G is cut into several connected subgraphs and source forwarding node $node_{\text{src}}$ and destination forwarding node $node_{\text{dst}}$ are in different subgraphs. Suppose there are $Q^n_{\text{src,dst}}$ sniffed node set V^n_{sniff} satisfying $V^n_{\text{sniff}} \supseteq V_{\text{cut}}$. Proof of the probability that attacker can obtain complete communication data in traditional network in one communication which is not less than that in DHC—that is, $P_{\text{traditional}} \ge P_{\text{hop}}$—is shown below.

Proof. Verify that $P_{\text{traditional}} \ge P_{\text{hop}}$; and make sure $P_{\text{traditional}} - P_{\text{hop}} \ge 0$.

Given $P_{\text{traditional}} = (C^n_m - C^n_{m-s})/C^n_m$, $P_{\text{hop}} = (C^1_2 C^{n-1}_{m-2} + C^2_2 C^{n-2}_{m-2} + Q^n_{\text{src,dst}})/C^n_m$, we have

$$P_{\text{traditional}} - P_{\text{hop}} = \frac{C^n_m - C^n_{m-s} - \left(C^1_2 C^{n-1}_{m-2} + C^2_2 C^{n-2}_{m-2} + Q^1_{\text{src,dst}}\right)}{C^n_m}. \tag{A.1}$$

Suppose the shortest path from H_1 to H_2 is $path^*$ ($path^* \in S^{H_1 \to H_2}_{\text{RH}}$). The complete communication data from source host to destination host can be sniffed on V^n_{sniff}; then $\forall path \in S^{H_1 \to H_2}_{\text{RH}}$, there exists $V^n_{\text{sniff}} \cap Nodes(path) \ne \varnothing$, where $Nodes(path)$ represents the set of nodes that $Path$ passes. Because $path^* \in S^{H_1 \to H_2}_{\text{RH}}$, then $V^n_{\text{sniff}} \cap Nodes(path^*) \ne \varnothing$; that is, V^n_{sniff} contains at least one node on the shortest path (*Conclusion 1*).

When $n = 1$, attack sniffs 1 node in the network. Then, based on (A.1), we have

$$P_{\text{traditional}} - P_{\text{hop}} = \frac{C^1_m - C^1_{m-s} - \left(C^1_2 C^0_{m-2} + Q^1_{\text{src,dst}}\right)}{C^1_m}. \tag{A.2}$$

In (A.2), the denominator $C^1_m > 0$ and the numerator is as follows:

$$C^1_m - C^1_{m-s} - \left(C^1_2 C^0_{m-2} + Q^1_{\text{src,dst}}\right) = m - (m-s) - \left(2 + Q^1_{\text{src,dst}}\right) = s - 2 - Q^1_{\text{src,dst}}. \tag{A.3}$$

Known by Conclusion 1, $V^1_{\text{sniff}} \cap Nodes(path^*) \ne \varnothing$; that is, the sniffed node is on the shortest path. In the s nodes on the shortest path, the number of V^1_{sniff} which can divide source node and destination node into different connected subgraphs is not more than $s - 2$; that is, $Q^1_{\text{src,dst}} \le s - 2$. So

(A.3) ≥ 0 can be got. The numerator of (A.2) is not less than 0; then, in (A.2) $P_{\text{traditional}} - P_{\text{hop}} \geq 0$.

When $n \geq 2$, attack sniffs more than 1 node in the network. Then, based on (A.1), we have

$$P_{\text{traditional}} - P_{\text{hop}} = \frac{C_m^n - C_{m-s}^n - \left(C_2^1 C_{m-2}^{n-1} + C_2^2 C_{m-2}^{n-2} + Q_{\text{src,dst}}^n\right)}{C_m^n}. \quad \text{(A.4)}$$

In (A.4), denominator $C_m^n > 0$ and the numerator is as follows:

$$\begin{aligned} &C_m^n - C_{m-s}^n - \left(C_2^1 C_{m-2}^{n-1} + C_2^2 C_{m-2}^{n-2} + Q_{\text{src,dst}}^n\right) \\ &= C_m^n - C_{m-s}^n - 2C_{m-2}^{n-1} - C_{m-2}^{n-2} - Q_{\text{src,dst}}^n \\ &= C_{m-2}^n - C_{m-s}^n - Q_{\text{src,dst}}^n. \end{aligned} \quad \text{(A.5)}$$

According to the definition, $Q_{\text{src,dst}}^n$ is the number of those V_{sniff}^n which can divide $node_{\text{src}}$ and $node_{\text{dst}}$ into different connected subgraphs. So $node_{\text{src}}$ and $node_{\text{dst}}$ do not belong to such V_{sniff}^n. C_{m-2}^n is the number of all V_{sniff}^n satisfying both $node_{\text{src}} \notin V_{\text{sniff}}^n$ and $node_{\text{dst}} \notin V_{\text{sniff}}^n$. $C_{m-2-(s-2)}^n$ is the number of V_{sniff}^n satisfying $V_{\text{sniff}}^n \cap Nodes(path^*) = \varnothing$. Known by Conclusion 1, $V_{\text{sniff}}^n \cap Nodes(path^*) \neq \varnothing$; then $Q_{\text{src,dst}}^n$ is not more than $C_{m-2}^n - C_{m-2-(s-2)}^n$. So (A.5) ≥ 0 can be got. The numerator of (A.4) is not less than 0; then, in (A.4) $P_{\text{traditional}} - P_{\text{hop}} \geq 0$.

In conclusion, $P_{\text{traditional}} - P_{\text{hop}} \geq 0$; that is, $P_{\text{traditional}} \geq P_{\text{hop}}$. □

Conflict of 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. 61379151, 61272489, 61302159, and 61401512) and The National Cryptography Development Fund of China (no. MMJJ201301005). The National Basic Research Program of China (973) (Grants nos. 2012CB315901 and 2013CB329104) and The National Natural Science Foundation of China (Grants nos. 61309019 and 61372121).

References

[1] National Cyber Leap Year Summit 2009 Co-Chairs' Report, "Networking and information technology research and development," Tech. Rep., 2009.

[2] 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.

[3] 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, New York, NY, USA, 2011.

[4] E. Al-Shaer, "Toward network configuration randomization for moving target defense," in *Moving Target Defense*, vol. 54 of *Advances in Information Security*, pp. 153–159, Springer, New York, NY, USA, 2011.

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

[6] M. Atighetchi, P. Pal, F. Webber, and C. Jones, "Adaptive use of network-centric mechanisms in cyber-defense," in *Proceedings of the 6th IEEE International Symposium on Object-Oriented Real-Time Distributed Computing*, pp. 183–192, Hokkaido, Japan, May 2003.

[7] 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 Workshop on Hot Topics in Software Defined Networks (HotSDN '12)*, pp. 127–132, ACM, Helsinki, Finland, August 2012.

[8] Q. Duan, E. Al-Shaer, and H. Jafarian, "Efficient Random Route Mutation considering flow and network constraints," in *Proceedings of the IEEE Conference on Communications and Network Security (CNS '13)*, pp. 260–268, IEEE, National Harbor, Md, USA, October 2013.

[9] E. Al-Shaer, Q. Duan, and J. H. Jafarian, "Random host mutation for moving target defense," in *Security and Privacy in Communication Networks*, pp. 310–327, Springer, New York, NY, USA, 2013.

[10] 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.

[11] H. Wang, Q. Jia, D. Fleck, W. Powell, F. Li, and A. Stavrou, "A moving target DDoS defense mechanism," *Computer Communications*, vol. 46, pp. 10–21, 2014.

[12] C.-Y. Hong, S. Kandula, R. Mahajan et al., "Achieving high utilization with software-driven WAN," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 3, pp. 15–26, 2013.

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

[14] M. Carvalho and R. Ford, "Moving-target defenses for computer networks," *IEEE Security & Privacy*, vol. 12, no. 2, pp. 73–76, 2014.

[15] M. Sifalakis, S. Schmid, and D. Hutchison, "Network address hopping: a mechanism to enhance data protection for packet communications," in *Proceedings of the IEEE International Conference on Communications (ICC '05)*, vol. 3, pp. 1518–1523, IEEE, Seoul, Republic of Korea, May 2005.

[16] M. Dunlop, S. Groat, W. Urbanski, R. Marchany, and J. Tront, "MT6D: a moving target IPv6 defense," in *Proceedings of the Military Communications Conference (MILCOM '11)*, pp. 1321–1326, IEEE, Baltimore, Md, USA, November 2011.

[17] J. H. Jafarian, E. Al-Shaer, and Q. Duan, "An effective address mutation approach for disrupting reconnaissance attacks," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 12, pp. 2562–2577, 2015.

[18] J. H. H. Jafarian, E. Al-Shaer, and Q. Duan, "Spatio-temporal address mutation for proactive cyber agility against sophisticated attackers," in *Proceedings of the 1st ACM Workshop on Moving Target Defense (MTD '14)*, pp. 69–78, Scottsdale, AZ, USA, November 2014.

[19] J. H. Jafarian, E. Al-Shaer, and Q. Duan, "Adversary-aware IP address randomization for proactive agility against sophisticated attackers," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM '15)*, pp. 738–746, IEEE, April 2015.

[20] D. C. MacFarland and C. A. Shue, "The SDN shuffle: creating a moving-target defense using host-based software-defined networking," in *Proceedings of the 2nd ACM Workshop on Moving Target Defense (MTD '15)*, pp. 37–41, ACM, Denver, Colo, USA, October 2015.

[21] 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., vol. 8134 of *Lecture Notes in Computer Science*, pp. 237–254, Springer, Berlin, Germany, 2013.

[22] S. Dolev and S. T. David, "SDN-based private interconnection," in *Proceedings of the IEEE 13th International Symposium on Network Computing and Applications (NCA '14)*, 2014.

[23] F. Gillani, E. Al-Shaer, S. Lo, Q. Duan, M. H. Ammar, and E. W. Zegura, "Agile virtualized infrastructure to proactively defend against cyber attacks," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM '15)*, pp. 729–737, Hong Kong, April-May 2015.

[24] D. Gkounis, V. Kotronis, and X. Dimitropoulos, "Towards defeating the crossfireattack using SDN," http://arxiv.org/abs/1412.2013.

[25] A. Studer and A. Perrig, "The coremelt attack," in *Computer Security—ESORICS 2009*, vol. 5789 of *Lecture Notes in Computer Science*, pp. 37–52, Springer, Berlin, Germany, 2009.

[26] 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*, ACM, October 2010.

[27] 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.

[28] M. McCauley, "About pox," 2013, http://www.github.com/noxrepo/pox/.

[29] S. De Maesschalck, D. Colle, I. Lievens et al., "Pan-European optical transport networks: an availability-based comparison," *Photonic Network Communications*, vol. 5, no. 3, pp. 203–225, 2003.

[30] A. Dixit, F. Hao, S. Mukherjee, T. V. Lakshman, and R. Kompella, "Towards an elastic distributed SDN controller," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4, pp. 7–12, 2013.