Polymorphic Path Transferring for Secure Flow Delivery

Rongbo Zhang[1,2], Xin Li[1*], Yan Zhan[3]
[1] State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications
Beijing, 100876, China
[e-mail: zhangrongbo@bupt.edu.cn]
[2] Zhengzhou University, Zhengzhou, 450001, China
[3]AVI-SPL, Burnaby BC V5A 4N4, Canada
[e-mail:yan.zhan@avispl.com]
[*]Corresponding author: Xin Li

Received January 23, 2021; revised June 6, 2021; accepted June 28, 2021; published August 31, 2021

Abstract

In most cases, the routing policy of networks shows a preference for a static one-to-one mapping of communication pairs to routing paths, which offers adversaries a great advantage to conduct thorough reconnaissance and organize an effective attack in a stress-free manner. With the evolution of network intelligence, some flexible and adaptive routing policies have already proposed to intensify the network defender to turn the situation. Routing mutation is an effective strategy that can invalidate the unvarying nature of routing information that attackers have collected from exploiting the static configuration of the network. However, three constraints execute press on routing mutation deployment in practical: insufficient route mutation space, expensive control costs, and incompatibility. To enhance the availability of route mutation, we propose an OpenFlow-based route mutation technique called Polymorphic Path Transferring (PPT), which adopts a physical and virtual path segment mixed construction technique to enlarge the routing path space for elevating the security of communication. Based on the Markov Decision Process, with considering flows distribution in the network, the PPT adopts an evolution routing path scheduling algorithm with a segment path update strategy, which relieves the press on the overhead of control and incompatibility. Our analysis demonstrates that PPT can secure data delivery in the worst network environment while countering sophisticated attacks in an evasion-free manner (e.g., advanced persistent threat). Case study and experiment results show its effectiveness in proactively defending against targeted attacks and its advantage compared with previous route mutation methods.

Keywords: Moving target defense, route mutation, software defined networking, segment routing.

This work was supported by Research on measurement and analysis technology of data center coverage quality (No. A2020227, Capital online Data Service Co., Ltd) and Open Foundation project of National Engineering Laboratory for Internet Medical System and Application (NELIMSA2018P04).

1. Introduction

In a network, to protect information transmission, some dedicated secure hardware and middleware are widely used. These devices play a critical role in guaranteeing the safe transmission of information. The additional equipment deployment increases the operating costs and investment costs. Thus, it confines the network expansion and limits the flexibility of the network service as well [1]. At the same time, most traditional tools and techniques can only defend against the known threats, and a few can cope with the latent network threats [2].

The static one-to-one mapping of the communication pairs to the routing paths offers adversaries a great advantage to conduct a thorough reconnaissance and organize an effective attack in a stress-free manner. Without adding dedicated devices, Random Route Mutation (RRM) has established itself as a powerful technique in coping with advance persistent threats and stealthy targeted attacks [3, 4]. RRM is one of the Moving Target Defense techniques. The core idea of RRM is that a network defender can proactively change the routing path of a communication pair to increase the complexity and costs of the attacks and decrease the attacks' successful probability of accessing the routing path [5].

In contrast to route mutation techniques, traditional secure sockets typically use encryption methods to forbid attackers from getting information from captured data. The static nature of the routing path is apt to lead to some advantages to attackers, such as sufficient time, prior knowledge of the routing paths [6]. Therefore, a well-crafted route mutation technique should have two evidence characteristics to degrade the successful attacks. The one is the high frequency of route mutation for decreasing the exploring time; The other is a large mutation space for enhancing the uncertainty and complexity of the routing path.

Adopting route mutation to deliver data of a pair in the network increases the amount of routing paths used in the communication. Some pathfinding algorithms have proposed to search for a routing path with more diversity for delivering data in the next time interval of the pair in the network. However, the effectiveness is limited by the network topologies. For example, the intersection exists between some paths of the routing path set. The non-disjoint routing path has little contribution to enhancing the security of the communication. At the same time, the routing paths should be meet some constraints (e.g., bandwidth, delay) to ensure the communication performance fulfilling the QoS of the flow.

It is often that most flows have an insufficient routing path space that decreases the effectiveness of route mutation. So, we introduce a polymorphic path transferring method based on assembling physic and virtual path splices. It generates flexibly routing paths for the next interval and enlarges the routing path space for routing mutation. Section 5 lists the capabilities of the method in detail.

To improve the effectiveness of route mutation, pattern recognition based on artificial intelligence has applied in route mutation in literature [7,8]. By introducing some reinforcement learning algorithms, the defenders can get more information about the attackers to adjust the time at changing the routing path. However, Advance Persistent Threats are still often invalidating these methods. To a defender, when to change the routing path is still a problem.

The design of route mutation has accelerated greatly by adopting Software Defined Networking architecture in which the intelligence of the network centralized in a logical controller. However, the scalability of the routing mutation is heavily affected by the control costs as well. Taking routing path mutation will bring additional control messages and storages on the data plane. There have been many different routing path update techniques available. But, routing mutation makes most of them inapplicable to a routing path update of high frequency due to unreasonable costs. So, the work proposed a segment path update to decrease the control and storage costs.

The main contributions of this work as follows:

1) A polymorphic path method with assembling physic and virtual path splices is introduced to enlarge the routing path diversity.

2) Route mutation optimization problem is modelled as a Markov Decision Process with uniform distribution of background flows, by value iteration to get optimal routing path at the mutation time.

3) Adopting a segment path update in which a path is updated piecemeal with aggregation flow entries in the forwarding table. It can decrease the control and storage costs.

The Polymorphic Path Transferring with Segment Path Update (PPT-SPU) technique can improve the adaptiveness and scalability of route mutation. The PPT-SPU allows the defender to implement effective routing mutation in an environment confined to network resources (e.g., insufficient routing path space, stressed control sources). It also enables the defender to execute the next-path decision and migration optimally. We empirically evaluate our scheme in an OpenFlow-based test network. Experimental results demonstrate the advantages of PPT-SPU over the other approaches in security and costs.

The remainder of the paper is organized as follows: Section 2 discusses the related works of route mutation. Section 3 presents the motivating case study, the problem formulation, and the security threats. Section 4 describes a corresponding model and a solution to the problem. some critical implementer points are given in Section 5. In Section 6, we conducted a case study, and effectiveness analysis and experimental results are demonstrated as well. Finally, we concluded the paper in Section 7.

2. Related Work

The defense measurements are usually launched passively when the harms occurred because the cyber attackers are often in the dark. To change the disadvantaged situation of the network defenders in the network attack and defense, Moving Target Defense (MTD) has attracted the attention of network researchers in recent years [3]. By proactively changing some of the network states, MTD can make an adversary in an ever-changing attack surface. So, this makes it harder for the attackers to determine the target of accuracy. MTD can respond effectively to known network threats. It can effectively reduce the security problems that may result from latent network threats also.

Since the traditional networks cannot meet the basic requirements of MTD that need to configure the network states dynamically at a high frequency, MTD confined in the domain of the single computer system for a long time, and few success cases in the Network Moving Target Defense (NMTD). In recent years, SDN (software-defined networking) has popularized as a network architecture [9]. With the wide adoption of SDN, the establishment of secure network services based on it has become one of the concerns [10]. SDN provides flexibility and programmability to the network managers and operators that make NMTD easier to implement.

Data transmitted in the network will face various kinds of attacks, such as Data Leakage [l1], Data Tampering [12], Data Replay [13]. Meanwhile, the development of machine learning in packet analysis makes the network transmission based on encryption face greater risks [14]. Therefore, limiting the number of data exposed to attackers will be an effective way to mitigate these attacks. different from the passive routing adjustment in case of network failure [15], The NMTD based on SDN has attracted more attention from academia and the industry of the network community in the last few years.

NMTD increases uncertainty/confusion to the adversaries by changing the network configuration continuously and dynamically. By now, the researchers have put more emphasis on changing static parameters of the data flow on the fly [16,17,18,19,20]. The IP address [21], the port [22], and the routing path of a flow [11,19,20,23,24,25] are the main objects of dynamic configuration.

Based on OpenFlow architecture, Jafarian et al., [18] proposed a flexible terminal IP faked data delivering method, called OF-RHM, and the analysis shows it can effectively decrease the successful attacks, but the mapped virtual IP keeps unchanged during the session. Gillani et al., [26] foil the adversaries' probing through migrating virtual routers among multiple paths, then resists the link DDoS attacks. Qi et al., [19] formalized the RRM problem to a Satisfiability Modulo Theories (SMT) to identify the constraint-satisfying routing path set of the mutating flow. It also pointed out that finding enough disjoint paths is a hard job in wired networks. They adopted overlay networking to expand the satisfaction of the routing path. However, creating and maintaining a virtual network is a high-cost work itself. Jafarian et al., [27] adopted a similar model to the literature [19], and the difference of their work is introduced the game theory into the optimal strategy selection. Both [19] and [27] are limited to how to mutate and not to when to mutate optimally.

Jiang Liu et al., [28] develop an optimal routing path selection algorithm by introducing a network anomaly trigger mechanism to cope with changing routing paths optimally. Duan et al., [29] propose an RM scheme to cope with Infrastructure Distributed Denial of Service. These schemes all consider the attack strategies of adversaries in deciding when to mutate. Some papers [7,8,30] have adopted Deep Reinforcement Learning to optimal routing mutation. The results show evidence effective on the optimal routing path selection and mutation time. With Artificial Intelligence development, it seems more effective techniques for searching optimal routing path and mutation time will occur.

From the literature, routing path diversity plays a critical role and decides the security effectiveness in routing mutation directly. Three main constraints execute press on the routing path diversity: capacity constraint, overlap constraint, and QoS constraint [19]. At the same time, fine-grained flow scheduling brings pressure to the flow table storage in the data plane [31]. Since changing the routing path with high frequency, we also need to consider the control cost constraint introduced by the central control model of SDN architecture.

In this paper, we propose a routing path expand method, called virtual and physical path slices consolidated, which is different from the overlay network in [19] and the other schemes that make full use of overlapped routing paths (e.g., 7,8,10). To search for an optimal routing path, a simple optimal algorithm with corresponding segment path update and Markov decision Process is used in solving the path scheduling. However, the advanced optimal algorithms, such as the Differential evolution algorithm [32], is expected to apply in route mutation with the improvement of the controller capacity in the near future.

In our experiment environment, we do not consider the optimal time to execute the mutation based on attacks' patterns due to a lack of trusted data from detecting. However, we hold the opinion that introduced intelligence attack sensor techniques in our scheme, which will enhance the defense instead of worsening it.

3. Problem Definition

3.1 Motivating case study

To resist some emergency, such as device invalid, link break, congestion, the operators often deploy redundancy resources in the network. It is a small probability that only one path exists between a communication pair in a running network. Parallel multipath brings opportunities at providing flexible routing policy to the data delivery of the communication pairs. By transferring data over different routing paths, route mutation enlarges the attack surface and decreases the exploration time of the adversaries. Therefore, it enhances the security of the fly packet in the network. The diversity of the routing path is a critical factor that decides the effectiveness of the route mutation. All simple routing paths of a communication pair construct a routing path set. There are two categories of these routing paths. One is disjoint paths, and the other is non-disjoint paths. The former proved more security effectiveness compared with the latter. It is satisfactory while having enough disjoint paths of a communication pair for executing route mutation policy.

We have extract ten network topologies from topology zoo [33]. These topologies are from the networks deployed around the world. Unfortunately, there are a few disjoint paths between the pairs in the topologies. There are many reasons to explain the cause of this case, such as operating costs, investment costs, and management costs. The statistical results showed in **Table 1**. O_N represents the overlap rate and N_N represents the number of nodes in all paths between a pair. The O_N is given by: O_N = (number of nodes in all paths - N_N) / N_N.

The overlay network is one of the techniques to meet the challenges. The operators can add and delete virtual links flexibly in the overlay network. Therefore, the amount of the routing paths of a communication pair can be expanded as needed. The main problem of the overlay networks is bringing extra management and maintenance costs. The optimal non-disjoint path application also brings some benefits to security. However, based on scarce path resources, it is only providing a limited effectiveness.

It is different from before when PPT-SPU adopts segment route and mixture routing path construction. For non-disjoint paths, the overlap part can be divided into multiple logical channels, called virtual path segments. Then, route mutation uses the routing paths reconstructed by the virtual path segments and the physical path segments. Compared with maintaining an overlay network, route mutation dynamically constructs the routing path at delivering data packets and tears down it at the end of delivery. So, a local virtualization-based routing path construction method has more flexibility and less cost.

Table 1. The Statistics of ten networks

Topologies	**N_N**	**O_N**
Aarnet	19	3.11
Biznet	28	4.11
Chinanet	38	1.51
Evolink	32	2.52
Forthnet	60	1.59
GtsCzechRepublic	26	4.4
GtsPoland	26	3.52
Litnet	39	2.05
OS3E	34	3.33
Rnp	29	2.89

3.2 Security Threats

Routers/switches are responsible for forwarding a flow in networks. However, backdoors of devices or poor security management introduce potential threats to data communications. There have many accidents created by attackers who compromised the routers/switches [34]. Therefore, in our works, some routers/switches are considered potentially malicious. That is, some routers/switches do not operate as expected.

An adversary seeks to compromise the node among the transferring path of the target communications pair, then based on the compromised node, they might execute directed or undirected attacks (e.g., eavesdropping, DDoS). The route mutation system schedules the packets of the flow from over one path to the others dynamically to escape the capture of attackers.

We assume the attacker is aware of the route mutation implemented by the communication pair in the network. The attacker selects a random path to explore the active flows of the target pair. The work assumed that an attacker has some level of ability to compromises a target path. Its capability as follows:

1) The attacker is aware of the network topology and the IP addresses of the communications pair via the reconnaissance and will attempt to compromise a route randomly between the communications pair.

2) Subject to a limited budget and an attempt to keep a low probability from detection, the attacker can only explore a portion of the nodes in the network in a fixed period. The attacker cannot monitor the whole network, which will let all effort used route mutation to lose efficacy.

3) The attacker can attack network nodes randomly at any time interval and moves from one path to the other while the target flow not identified.

4) The attack can choose to attack a set of nodes based on the knowledge increases over time.

5) If the attacker hits the flow of the target communications pair (the attacker hits the node in the routing path), it will stay in the state till the flow disappear.

3.3 Problem formulation

For transferring data from source node S to destination node D in a network, there is often a group of simple paths between S and D, we denoted it denoted by P^{SD}. At the same time, we assume that P^{SD} has more than one element (it is a common situation in a robust network), and $P^{SD} = \{P_i^{SD} | \; i \in [1 \; , \; n^{SD}]\}$, where P_i^{SD} indicates the ith path of the path set, and the element number of the P^{SD} is $n^{SD}=|P^{SD}|$. A routing path consists of a sequence of network nodes. We let $R_{i,k}^{SD}$ represent the kth node of the path P_i^{SD}. The main notations and definitions are shown in Table 2.

Table 2. Portion notations and definitions

Notations	Definitions
P^{SD}	The set of simple paths between source node S and destination node D
P_i^{SD}	The ith path of the path set P^{SD}
n^{SD}	The number of elements in the P^{SD}
$R_{i,k}^{SD}$	The kth node of the path P_i^{SD}
P_H^{SD}	The routing path set of the pair <S, D> while adopting route mutation
$A_{i,k}^{SD}(\mathrm{t})$	The cumulative number of flows arriving at $R_{i,k}^{SD}$ in the time interval $[0, t]$
$A_{i,k}^{SD}(\tau, \mathrm{t})$	The number of flows arriving at $R_{i,k}^{SD}$ in the time interval $[\tau, t]$

$D_{i,k}^{SD}$(t)	The cumulative number of flows that have departed from $R_{i,k}^{SD}$ during the time interval $[0, t]$
$E_{i,k}^{SD}$(τ, t)	The existing flows processed at $R_{i,k}^{SD}$ during the period of (τ, t)
AS_i^{SD}	The attack surface of the routing path P_i^{SD}
AS^{SD}	The attack surface of the flow delivered form S to D adopted route mutation

In a classic packet transfer from S to D, it often uses the shortest path, and only one routing path of P^{SD} is used to deliver all packets of the flow. That is, the transmission is over the network nodes along the shortest path. However, adopting route mutation, packets of a flow will fly over different routing paths, and we denote the used routing paths by P_H^{SD} ($P_H^{SD} \subset P^{SD}$). The packet distribution on the P_H^{SD} is relative to the path scheduling algorithm.

For kth router (or switch) $R_{i,k}^{SD}$ in the path P_i^{SD}, the cumulative number of flows arriving at it in the time interval $[0, t]$, we denote by $A_{i,k}^{SD}$(t), and the value is non-negative. To formulate the amount of arrived flow precisely, the notation $A_{i,k}^{SD}$(τ, t) represents the amount of arrived flows in the time interval $[\tau, t]$, where $t > \tau \geq 0$. Combined with the definition of $A_{i,k}^{SD}$(t), $A_{i,k}^{SD}$(τ, t) is expressed as follows:

$$A_{i,k}^{SD}(\tau, t) = A_{i,k}^{SD}(t) - A_{i,k}^{SD}(\tau)$$
$$\text{Where } A_{i,k}^{SD}(t, t) = 0, \forall t > 0. \tag{1}$$

The symbol $D_{i,k}^{SD}$(t) used to denote the cumulative number of flows that have departed from $R_{i,k}^{SD}$ during the time interval $[0, t]$. Since the number of departed flows cannot exceed the amount of arrived flows, so, $A_{i,k}^{SD}$(t) > $D_{i,k}^{SD}$(t), ∀t > 0. In particular, when $A_{i,k}^{SD}$(t) > $D_{i,k}^{SD}$(t), there are many flows that are lively at $R_{i,k}^{SD}$.

For a router/switch, a flow commonly consists of a sequence of packets needed to forward to the next node. To associate $A_{i,k}^{SD}$(t) with $D_{i,k}^{SD}$(t), and we assume that each flow is time-confining, that is, the flow does not live on a node if there are no packets to be transferred. The existing flows processed at $R_{i,k}^{SD}$ during the period of (τ, t) is denoted by $E_{i,k}^{SD}$(τ, t), where $E_{i,k}^{SD}$(0, t) = $E_{i,k}^{SD}$(t). So, given $R_{i,k}^{SD}$ and two-time points τ and t (t > τ), we can get:

$$E_{i,k}^{SD}(\tau, t) = A_{i,k}^{SD}(t) - D_{i,k}^{SD}(\tau) \tag{2}$$

Given a path P_i^{SD} used to transfer data for pair <S, D>, and the duration of the flow is the interval (t_1, t_2). Thus, in the worst scenario, the attacker is only need to explore one router/switch among the path, and which has the minimum number of flows under process comparing with the other nodes among the routing path. The attack surface of the routing path P_i^{SD} is denoted by AS_i^{SD}, and its value is:

$$AS_i^{SD} \geq min_{R_{i,k}^{SD} \in P_i^{SD}} E_{i,k}^{SD}(t_1, t_2) \tag{3}$$

To increase security, route mutation scales up attacker surface at detecting a flow on the flying. For a flow of pair <S, D>, if all paths in the P^{SD} are disjointed, the attack surface of the flow (denoted by AS^{SD}) is formalized as:

$$AS^{SD} = \sum_{P_i^{SD} \in P_H^{SD}} AS_i^{SD} \tag{4}$$

To update a path P_i^{SD} (consisting of $|P_i^{SD}|$ nodes) in OpenFlow networks, under the classical scenario, it will bring $|P_i^{SD}|$ Flow-Mod messages at the initial phase of the flow transfer and $|P_i^{SD}|$ Flow-Removed messages at the finish phase. The storage requirement is $|P_i^{SD}|$ flow entries on the data plane.

For a path replacement, such as using P_j^{SD} to replace P_i^{SD}, the extra $|P_j^{SD}|$ Flow-Mod messages and $|P_j^{SD}|$ Flow-Removed messages add to exchange messages between the controller and the data plane. It also brings an instant increase at storage on the data plane. To assure the packets delivery at path exchange period, it is need to store $|P_i^{SD}|$+$|P_j^{SD}|$ flow entries on the data plane in a short period.

Traditionally, in the shortest path first routing paradigm, a flow overhead of pair <S, D> includes: 2*$|P_i^{SD}|$ + 1 control messages and $|P_i^{SD}|$ flow entries storage, where P_i^{SD} satisfied $|P_i^{SD}| = min_{P_j^{SD} \in P^{SD}}|P_j^{SD}|$.

For route mutation with frequency H, due to routing path replacement constantly, a flow overhead of the pair <S, D> duration time T includes: $2 * \sum_{k=1}^{TH}\left|P_k^{SD}\right| + 1$ control messages (denoted by $C_{control}^{SD}$) and peek storage requirement for $|P_k^{SD}|$+$|P_{k+1}^{SD}|$ flow entries (denoted by $C_{Storage}^{SD}$). Where $|P_k^{SD}|$ and $|P_{k+1}^{SD}|$ are the two paths in a temporally alternating sequence and P_k^{SD} is the kth time interval used path.

Therefore, in route mutation, path P_j^{SD} used for replacing the current path for delivering the flow of pair <S, D> involves some control messages of Flow-Mod and Flow-Removed and the burst of storage. When the routing path mutation is at a fast frequency that will bring high operation overhead on control message. At the same time, the data plane requires more storage coping with changing paths at high frequency. All in all, route mutation brings more pressure on the scalability.

For improve the scalability of route mutation, decreasing the overhead on the path replacement ($C_{control}^{SD}$ and $C_{Storage}^{SD}$) is a critical way. Based on the SDN architecture and OpenFlow characteristics. In PPT-SPU, network background flows enter in consideration to seek a strategy that gets:

$$\min\left(C_{control}^{SD} + C_{Storage}^{SD}\right) \quad and \ \max(AS^{SD}) \tag{5}$$

4. Model Description and Solution

Considering a periodic-mutation route communications system with control messages, storage in data plane, quality of service (QoS) and security, we model the problem as a Markov Decision Process (MDP) to obtain optimal mutation policy. The transferring flows over periods $\{F_1, F_2, \dots, F_t, \dots\}$ are independent and identically distributed (iid) between periods in the network. At each time $t = 1,2, \dots$, the F denote one-period transferring flows, which is non-negative with mathematical expectation $E[F] > 0$, and the defender has not access to the true transferring flow in the network a priori. The defender can only observe the past statistics of flows and adjust the routing path on the fly.

4.1 Markov Decision Process Model

We model the problem as a MDP to obtain optimal mutation policy. There are two mutation options as follows:

1) “virtual” segment: Based on changing the identifier of the flow to construct a logical channel, it can escape the capture of the attack among the path. The significant feature is its diversity.

2) “physical” segment: Not only the identifier of the flow but also the routers/switches among the routing path are all changed while adopting “physical” segment mutation. However, the diversity is often insufficient.

We can obtain revenue from changing the identifier of the flow, varying the routers/switches among the routing path, or both in forwarding the flow, which can increase difficulty of the attackers. However, the costs are also paid to migrate the routing path or identifier of the flow. We concentrate the overhead on control messages and storage of the flow entries in here.

The routing path state migrating decisions are made once a time interval. The decision is based on the observations that includes current forwarding flow state (the identifiers of the flow and routers among the routing paths). Depending on the network states, the decided actions at the delivering flow make more sense than other. And the routing path should also meet:

1) No loop in the routing path;

2) Can’t load beyond any link or node capacity among the path;

3) QoS of the flow.

Each action of route mutation has a reward which equals the revenue subtracting the costs. Taking an action will affect what we observe next. That is the next period’s routing state of the flow, and it includes amounts of flows forwarded at each router among each routing path.

For the routing mutation system under consideration, at each time $t = 1,2, \ldots$, it observes the current routing state $x_t \in X$. It will select a mutation action for transforming the routing path in next period $a_t \in A(x_t)$, which will earn a reward $r(x_t, a_t)$. The next routing state is x_{t+1} with probability $pr(x_{t+1}|x_t, a_t)$.

The action is periodically procuring a single path segment to migrate the routing path. The flows are independent and identically distributed (iid) among the routers in the network. The routing path set of pair <S, D> is $X = P^{SD}$. The action set is $A(x) = \{\text{virtual segment muation, physical segment mutation}\}$, which select path segments $ps, ps \in p_i,\ p_i \in P^{SD}$ } for delivering flow in the next time interval. We let $r(x, a)$ be the expected reward earned when action a is taken in state x:

$$r(x, a) = AS^{SD} - C^{SD}_{control} - C^{SD}_{Storage} \tag{6}$$

Let F_r denote the distribution of transferring flow at router r. If the current routing state is x, and action a is taken, the next state is x' in the next interval with probability:

$$pr(x'|x, a) = \left(\frac{\sum_{r\in x'} \int dF_r}{|x'|}\right) / \sum_{p_i \in P^{SD}} \left(\frac{\sum_{r\in p_i} \int dF_r}{|p_i|}\right) \tag{7}$$

Based on routing states, the defender follows a policy π for selecting actions. If the routing state history of the flow up to time t is $x_0 \ldots x_{t-1}a_{t-1}x_t$, then select action $a_t \in A(x_t)$ with probability $\pi_t(a_t|x_0 \ldots x_{t-1}a_{t-1}x_t)$, In general, policies may be randomized and history-dependent.

4.2 Solution

The performance of the route mutation measures by an Infinite-Horizon Discounted Total

Reward, denoted by $V_\gamma^\pi(x)$, which represents the expected value of the received rewards, starting from state $X = x$ and following policy π.

E_x^π = expectation given that the initial state is x and the policy π is used;
γ = discount factor (between 0 and 1);
X_t = routing state at time ;
A_t = migrating action taken at time ;

$$V_\gamma^\pi(x) = E_x^\pi \sum_{t=0}^{\infty} \gamma^t r(X_t, A_t) \quad x \in X \tag{8}$$

In route mutation, a deterministic stationary policy π is used to implement the mapping function: $: X \rightarrow$ A, which dictates each migrates to take a certain segment while being in a specific routing state. $\pi_x = a$ represents adopting action a at routing state x by using policy π. The state process $X_0, X_1, \ldots$ is a homogeneous Markov chain. Each routing migration rewards $r(x, a)$ are bounded. The policy π can be evaluated by the utility function as follows:

$$V_\gamma^\pi(x) = \sum_{x' \in X} pr(x'|x, \pi_x)[r(x', \pi_x) + \gamma V_\gamma^\pi(x')] \tag{9}$$

Since the routing state and action sets are finite. For any $\gamma \in [0,1)$, there exists an optimal policy that is deterministic and stationary. (Theorem, Howard, 1960). Consider any $\gamma \in [0,1)$, the value function:

$$V_\gamma^*(x) := {}^{inf}_{\pi} V_\gamma^\pi(x) \tag{10}$$

Uniquely satisfies the optimality equation

$$V_\gamma^*(x) = \max_{a \in A(x)} [r(x, a) + \gamma \sum_{x' \in X} pr(x'|x, a) V_\gamma^\pi(x')], \quad x \in X \tag{11}$$

Moreover, a deterministic stationary policy π is optimal if and only if

$$\pi^*(x) = \arg\max_{a \in A(x)} [r(x, a) + \gamma \sum_{x' \in X} pr(x'|x, a) V_\gamma^\pi(x')] \tag{12}$$

Based on the preceding the following algorithm, it can be used to compute an ε-optimal policy, i.e., a policy π^* such that:

$V_\gamma^{\pi^*}(x) \geq V_\gamma^*(x) - \varepsilon \;\; \forall x \in X$, for any $\varepsilon > 0$.

Then, we use the value iteration to solve the MDP problem, as shown in **Algorithm 1**.

Algorithm 1. Value Iteration

```
1: Select any $\varepsilon > 0$. Set $V_0^\pi(x) = 0, \forall x \in X$.
2: Set Δ=0, k=0
3: do
4:         for each $x$ in $X$
5:                 $V_{k+1}^\pi(x) = \max E_x^\pi[r(X_0, A_0)] + \gamma E_x^\pi[V_k^\pi(X_{t-1})]$
6:                 $\Delta = \max(\Delta, |V_{k+1}^\pi(x) - V_k^\pi(x)|)$
7:   Set k=k+1
8: while Δ < ε
9: return any policy the policy $\pi^*(x)$
```

5. Critical Implementer Points

The fundamental modules of PPT-SPU are the routing path calculator and scheduler. Firstly, based on the K-shortest paths algorithm from [35], the routing path calculator produces a simple path set for a route request from a communication pair. Each element in the set is a candidate to transfer data of the communication pair. A routing path consists of a sequence of network nodes (switches/routers). These devices are responsible for transmitting packets of the flow from the source to the destination terminal of the communication pair. In the network, each switch should be responsible for processing different flows at the same time. Both the monitor for flows matrix and the K-shortest paths selection algorithm are all outside the scope of this paper, and the reader can get these from our previous work [36]. Subsequently, the routing path scheduler that consists of path selection, path install, and path replacement enters the role of maintaining the routing path used to deliver the flow. We focus our attention on path selection and update in route mutation to attain optimal scalability and security performances as formulation (5) described. In the rest of the paper, the router, the switch, and the node would be interchangeable when no cause confusion.

As a route mutation policy for SDN networks, PPT-SPU implements path scheduling based on the global network view provided by a logical controller of the network. For adjusting a routing path, the central controller downloads instructions into the network device in the data plane. The PPT-SPU above the controller is responsible for flexibly scheduling the routing path of the flow on the fly.

The scalability of route mutation will seriously degrade while the costs severe increases caused by changing the routing paths at high frequency. For better scalability, the PPT-SPU adopts a segment path migrating strategy according to virtual/physical segment construction of the routing path. Then, the routing path migrates at a sequence of the path segments. The approach relieves the controller press caused by updating the whole routing path once a time. As a by-product, the control consumption (such as control bandwidth, storage) does not incur burst, and the communication performance does not intermittently deteriorate.

For further concealing the packets of the flow, during the virtual segment path construction and migration, it takes the network background flows into account at deciding the next-hop routing path, which introduced not only as a transfer performance index but also as a security index. The flow identifier changes according to the background flows that confuses the attacker in recognizing the target flow. So, it will further increase the security of flow delivery.

5.1 Path scheduler

In PPT-SPU, for changing the routing path of a flow on the fly, the path scheduler should decide the next-hop routing path for transforming the route of the flow from the current to the next. To attain optimal security with the least costs, it adopts MDP with a segment update strategy to determine the routing path for the next transmitting slot. The constraints of the next-hop path selection include the migrating cost, the traffic of the flow distribution history, and the network background flows. To reduce the control costs, the strategy also constructed virtual path segments by existed flow entries in the data plane, which provides better scalability. At the same time, integrating the flow identifier concealment, the security of data transfers is also enhanced.

Enlarging the explore space of the attackers is important for implementing a successful route mutation. The flows processed on different network nodes consist of the explore space. A router with a large amount flows at processing will add hard to the attacker at identifying the targe flow. So, the scheduler prefers to use the busy router among the routing path in the next time interval.

A routing path weight index is introduced to indicate how security the routing path is. We assume that the nodes of the network have the same capacity in processing of flows, the weight value of path p_i^{SD} in time slot (τ, t) is the average amount of the existing flows processed at the nodes among the path, we denoted it by W_i^{SD}.

$$W_i^{SD} = \frac{\sum_{1<k<|P_i^{SD}|} E_{i,k}^{SD}(\tau,t)}{|P_i^{SD}|} \tag{13}$$

Since the network background flow is a continuously changing state, the value of the weight is varying along with the time.

In SDN networks, an OpenFlow-enabled switch forwards the arrived packet according to the instructions of the matching flow entries in its flow table. At the initial phase, the matching field can be select from 12 metadata in OpenFlow 1.0 [37], and the number has extended to 42 metadata in OpenFlow 1.5 [38]. But, in practice, the number of the matching field in a flow entry is just a few. While two flows have a common path segment, their flow entries among the segment will have the common instructions. So, the flow entries can be multiplexed by both flows. Flow entries multiplex decreases the operation on both controller and data plane. It is also the major motivation for the proposed Segment Path Update.

As **Fig. 2** shows, a flow of pair <A, B> is transferring in the network, a path ($P_i^{AB} = (r_1, r_3, r_4, r_5, r_6)$) serviced the flow. When a new flow of pair <S, D> is occur, and the first packet of the flow arrives at r_2, then path P_j^{SD} is selected by the routing algorithm for transferring the flow. The path $P_j^{SD} = (r_2, r_3, r_4, r_5, r_7)$. Since $P_i^{AB} \cap P_j^{SD} = (r_3, r_4, r_5)$, segment (r_3, r_4, r_5) is multiplexed by the two flows. To benefit the multiplexing of the segment, at the ingress of the segment, the head of packets of the flow <S, D> can be modified to a new head which contains all the matching fields of the entries serviced the flow <A, B>. To distinct and restore the flow of <S, D> at the egress of the segment, an extra field needs to add in the head of packets of the flow <S, D>. Thus, the packets of the flow <S, D> are replaced at r3 and restored at r5. So, on the other nodes of the segment, the packets of the pair <S, D> will be processed as the packets of the pair <A, B>.

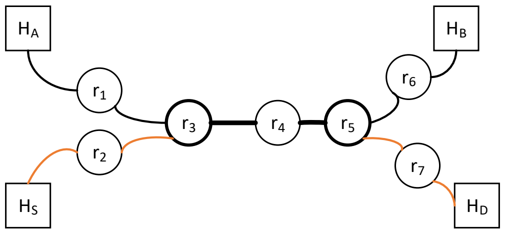

Fig. 2. The multiplexing of segment

In route mutation, virtual segment of a path implementation is to construct a virtual channel for the packets of the flow. In the virtual channel, it is need to cast the packets of the flow to transfer according to the special virtual channel. Finding an appropriate sequence of segments benefited to multiplex is the first task due to faking the flow is important for both virtual segment construction and Segment Path Update. There are different segment discovery paradigms (e.g., minimize the number of the segments contained in a path). Given the constraints, the transfer order of switches and segments in each path will be determined. Subsequently, the controller needs to execute the flow head replacement to disguise the flow for transmitting over a specified segment. To relieve the operation cost, minimize the number of the segments contained in a path is adopted in PPT-SPU.

Path P_i^{SD} can be divided into many segments. Conversely, the path consists of some segments which are stick to each other according to the directed sequence. For any segment in path P_i^{SD}, the start switch of the segment initials the packets transmitting of a flow over the segment, and at the end switch of the segment finishes the current segment transmitting and opens the next segment transfer (if it is not the end of the path).

We formally define the problem of segment discovery on a path set of pair <S, D> denoted by P^{SD}. We denote the set of all possible segments of P^{SD} by a segment dictionary S^{SD}, and each element in S^{SD} is a sub-path of a path in P^{SD}. It is easy to see that if path P_i^{SD} contains n_i^{SD}-1=|P_i^{SD}|-1 edges, then S_i^{SD} contains |S_i^{SD}| = (n_i^{SD} (n_i^{SD}-1))/2 segments.

The paths in P^{SD} can be reconstructed by concatenating the segments in dictionary S^{SD}. The size of the segment dictionary is denoted by |S^{SD} | which defines the number of its segments. Obviously, the largest segment dictionary that one can have is $S_{max}^{SD} = \bigcup_{i\in[1,n^{SD}]} S_i^{SD}$. When path P_i^{SD} can be reconstructed by a segment set $S_i^{SD\prime} = (S_1, S_2, \dots, S_m) \subset S_{max}^{SD}$, we denote by $P_i^{SD} = U(S_i^{SD\prime})$, there exists a permutation σ=σ(1) σ(2) … σ(m) such that $P_i^{SD} = S_{\sigma(1)} \cup S_{\sigma(2)} \cup \dots \cup S_{\sigma(m)}$.

Deduction. 1. If each path in P^{SD} can be reconstructed by two segments S_1 *and* S_2, and we denoted it by $(S_1, S_2) \subset S^{2SD}$, therein S^{2SD} represents the segment dictionary that each element in the S^{2SD} can find the other one constructed at least one path in the P^{SD}. That is $S^{2SD} \subset S^{SD}$, any path $P_i^{SD} = U(S_1, S_2)$, and there exists a permutation σ such that $P_i^{SD} = S_{\sigma(1)} \cup S_{\sigma(2)}$. We represent the segment $S_{\sigma(1)}$ and $S_{\sigma(2)}$ by $S_{(S,I)}$ and $S_{(I,D)}$ respectively, and node I is the finish node of $S_{\sigma(1)}$ and the start node of $S_{\sigma(2)}$.

Note: Given path P_i^{SD} and a segment dictionary S^{2SD}, there may exist multiple ways to reconstruct P_i^{SD} using subsets S^{2SD}. In our works, we are interested in the S^{2SD} for gaining largest attack surface.

Deduction. 2. The Attack Surface $AS(P_i^{SD}, t, S^{2SD})$ of a routing path P_i^{SD} transferring a flow at interval$(0, t)$ with respect to a segment dictionary S^{2SD} is given by:

$$AS(P_i^{SD}, \mathrm{t}, S^{2SD}) = AS_{S_1}^{SD} + AS_{S_2}^{SD}$$
$$s.t. \exists\, S_{sub} \subset S^{2SD},\; P_i^{SD} = \mathrm{U}(S_{sub}), S_{sub} = (S_1, S_2), S_1 = S_{(\mathrm{S,I})} \textit{ and } S_2 = S_{(\mathrm{I,D})}. \quad (14)$$

For segment discovery, we would like to select the segments to maximize the attacker surface of the extracted segment, as well as the largest average number of processed flows at network nodes along with the segments.

Given path set P^{SD} and the set of all possible segment S^{SD} , we define segment discovery as an optimal choosing repeatedly process of each path in P^{SD}. For ith path in P^{SD}, the discovery process as follows:

$$\text{Discovery}\,(P_i^{SD}, t,\, S_{sub}) = max_{S_{sub}\subset S^{2SD}, S_{sub}\notin S_{opt}^{2SD}, P_i^{SD}=\mathrm{U}(S_{sub}), P_i^{SD}\in P^{SU}} \mathrm{AS}(P_i^{SD}, \mathrm{t}, S^{2SD})$$
$$s.t. \forall\, P_i^{SD} \in P^{SU}, \exists S_{sub} \subset S^{2SD},\;\; P_i^{SD} = \mathrm{U}(S_{sub}) \quad (15)$$

Each segment discovery process will find a S_{sub} , if the S_{sub} exists then $S_{opt}^{2SD} = S_{opt}^{2SD} \cup S_{sub}$, else the segment discovery finishes and stops the next search. The segment set S_{opt}^{2SD} is the segment dictionary for the two-segment path replacement. Then, based on **Algorithm 1**, the selected routing path P_i^{SD} for the next interval is decided.

As shown in **Fig. 3**, when the controller received a packet of the route mutation service, it delivers the message to the Path Scheduler. The Path Scheduler calls the MDP module to select an optimal policy from the Policy set. The policy can be virtual segment mutation or physic segment mutation according to the reward. Then the Path Scheduler decides the routing path and produces the Flow-Mod Messages which are download into the data plane through the controller. The flow entries contained in the Flow-Mod Messages are installed into the switches among the routing path, then the packets of the flow will be transferred by the path. When the timer of the time interval expired or a threat even arrived, the path scheduler will start a new routing path production and paving process.

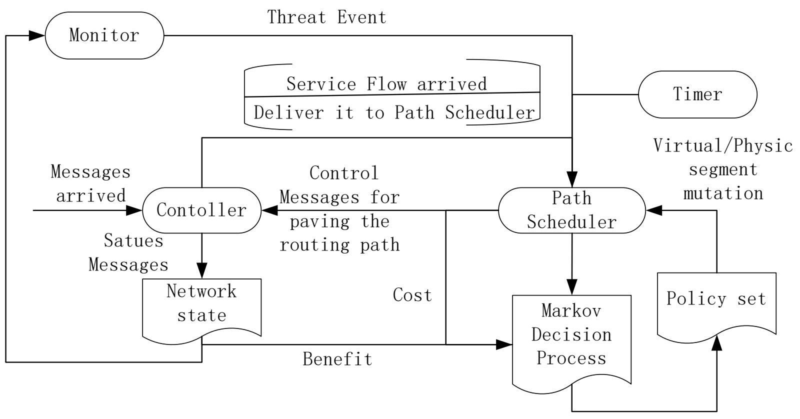

Fig. 3. The workflow of the PPT-SPU

In PPT-SPU, the optimal strategy selection algorithm adopted in MDP is value iteration, as **Algorithm 1** described. The time complexity of PPT-SPU is mainly in value iteration, which is O(nk), where n is the number of the policy and the k is the convergence steps. Besides, calculate reward and adjust the transition probability also caused time consumption. The storage consumption is mainly caused by intermediate results.

5.2 Virtual segment implementation

To implement virtual path segment, in PPT-SPU, a routing path is selected according to the **Algorithm 1**, and two segments S_1, S_2 from S_{opt}^{2SD} are determined ($S_1, S_2 \in S_{sub}$). To construct virtual segment based on S_1 or S_2, a shield channel should to be chosen carefully. A shield channel is a serial of flow entries that has paved by other flows, which can complete packets transport over one segment.

A flow set F contains some flows, such as $f_1, f_2, \dots, f_n$ ($F = (f_1, f_2, \dots, f_n)$), and a flow contains a serial of packets, then the number of the packets of the flow f_i is $|f_i|$. We use F_{S1} and F_{S2} to represent the flows over S_1 and S_2 respectively. To better conceal the flow, the selected channels are paved by the flow f_{k1}that satisfied $|f_{k1}| = max_{\,f_i \in F_{S_1}} |\, f_i|$ s.t. $f_{k1} \in Fs1$, and the flow f_{k2} that satisfied $|f_{k2}| max_{\,f_j \in F_{S_2}} |\, f_j|$ s.t. $f_{k2} \in F_{S2}$. Then flow f_{k1} and f_{k2} are the masking flows for segment S_1 and S_2 respectively. That is, the flow disguised as f_{k1} over S_1, then disguised as f_{k2} over S_2, and restored at the egress of S_2 the flow.

At a path updating, there are only three controller messages that need to be created and downloaded corresponding flow entries into the access node, the intermedia node, and the departure node of the path for the two-segment replacement approach. The intermedia node is the end node of the S_1 and the start node of the S_2. Base on the installed flow entries, when a packet of flow f arrived, at the access node the identifier of the packet is changed from f to f_{k1} identifier + extra fields for multiplexing S_1 with f_{k1}, then at the intermedia node the identifier of the packet is substituted with f_{k2} identifier +extra fields for multiplexing S_2 with f_{k2}, and at the departure node the identifier of the packet is restored to f, then the packet arrived the destination, the effects of the anonymity of the identifier disguise can enhance security of the flow on the fly.

5.3 Security Measurement

In route mutation application, it assumed that the switches in the networks are vulnerable spots of the network. The adversaries target a special flow for a long time. The betweenness centrality C_B is introduced to metric the security of the flow on the fly. Assumed a network consisting of vertexes and valued edges, the betweenness centrality of vertex i is the fraction of geodesic paths between other vertices that i falls on [39]. So, while the flows are independent and identically distributed (iid) among the routers, the high C_B of a vertex means exist many shortest paths of pairs across it. Since the classic routing rule of the network is the shortest path first, so C_B is an appropriate metric to measure the security of the threat model described in section 3 and the decision model in section 4. A compromised switch (vertex) with high C_B will bring substantial confusion to the attacker due to more flows over it.

Given a flow of pair <S, D>, and a path set P^{SD} of the flow, which can be used to transfer the flow. the security analysis as follows:

Shortest Path First: In this model, path p_s is the selected path from the P^{SD}, where $p_s = min_{P_k^{SD} \in P^{SD}}(\sum_{(i,j)\in P_k^{SD}} weight(i,j))$. We define $n_i^{SD} = |p_s|$, $p_s = (r_{\sigma(1)}, r_{\sigma(2)}, \dots., r_{\sigma(n_i^{SD})})$, that is, path p_s contains switches $r_{\sigma(1)}$, $r_{\sigma(2)}$,…, and $r_{\sigma(n_i^{SD})}$. So, the lowest betweenness centrality of the switch in path p_s (denoted by $C_{B,s}^L$) represents the worse scenario of security, $C_{B,s}^L = min_{i\in(1,n_s^{SD})} C_{B,r_{\sigma(i)}}$, and the security of Shortest Path First is $C_{B,s}^L$.

Route Mutation: we assume paths P_H^{SD} are used to transmit the flow, where $P_H^{SD} \subset P^{SD}, n_H^{SD} \leq n^{SD}$. For the paths of P_H^{SD} are disjoint , the lowest betweenness centrality of the switch in different paths of P_H^{SD} is different. So, the security of route mutation is $C_{B,H}^L = \sum_{p_i^{SD}\in P_H^{SD}} C_{B,i}^L$. For the paths of P_H^{SD} existing overlap, and if the overlap is the lowest betweenness centrality of many paths, the security of route mutation needs to guarantee no repeated calculation.

PPT-SPU: It assumed H paths P_H^{SD} are used to transfer a flow, where $P_H^{SD} \subset P^{SD}, n_H^{SD} \leq n^{SD}$. For the paths of P_H^{SD} are disjoint , each path in P_H^{SD} is divided into two sub-paths (segment) with different flow identify. Give any path $p_i^{SD} \in P_H^{SD}$, each sub-path of p_i^{SD} has the lowest betweenness centrality, and the value is not lower than $C_{B,i}^L$. So, the security of Two-Segment Update is $C_{B,R}^L \geq 2 * C_{B,H}^L$. Similar to route mutation, For the paths of P_H^{SD} existing overlap, the value needs to guarantee no repeated calculation.

From the analysis, we can see that the more path used to transfer a flow, the more security acquisition, the more segments in path updating, the more security acquisition. However, more paths and more segments mean more overhead which will be illustrated in the next section.

6. Performance evaluation

When a new flow arrived the network, it initials a flow routing process. That is, when a packet of the flow arrived at the access node, a path is selected by the controller according to the route strategy, and corresponding rules are installed into the data plane for transferring the flow. Next, it will change the path if need be, a new path will be selected, and the flow entries of the path will be downloaded into the corresponding switches along the path again.

As for two-segment path update with the frequency H, duration time T, the overhead for a flow of pair <S, D> contains $3 \times T \times H + 1$ control messages and peak storage requirement for $2 \times 3 = 6$ flow entries. Because, in two-segment replacement, only source, intermediate, and destination need to acquire the corresponding flow entries for path deployment. The two-segment path update can decrease overhead obviously on the storage requirement in the data plane and control messages which is important for the scalability of route mutation.

In the next section, we presented the results of a demo application that adopts PPT-SPU to enhance the security of a communication pair. This application is implemented based on the OpenFlow three-layer architecture, as shown in **Fig. 4**.

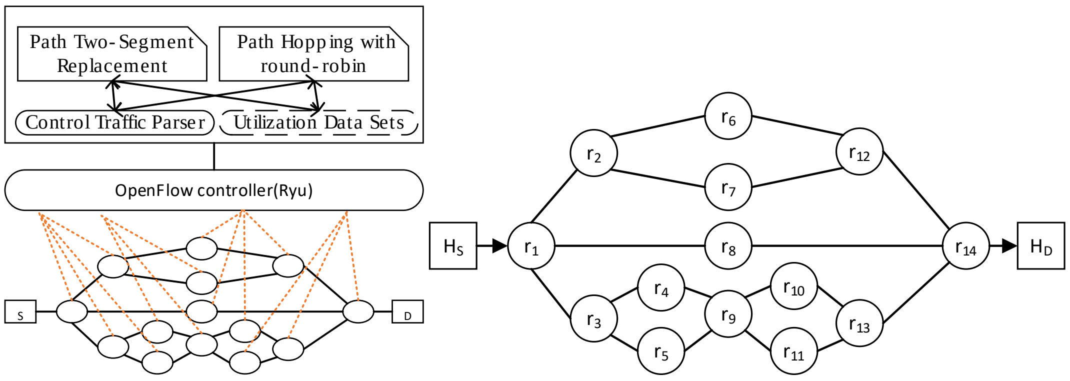

Fig. 4. The application architecture and the network topology for experiments

6.1 Experiment configuration

To verify the efficiency and effectiveness of PPT-SPU, A simple network was deployed in Mininet [40]. The network consists of 14 forwarding devices (Open vSwitch [41]) and a controller (Ryu [42]), and each switch connected with one host, as shown in **Fig. 4**. We set the bandwidth of each link in the network with 10Mbps. The packet delivery rate is 1000 per second at terminal S. The betweenness centrality C_B of each node is a random variable in range (300,1000). The other flows come from ping each other of the hosts, and we assume that is no traffic congestion in the network during the experiment. We implemented a base scenario, in which, to protect packets of the specified flow on the fly, the application periodically changes the paths at a constant interval. As **Fig. 4** shown, the disjoined paths from S to D is three. A common tool $nping$ is used to generate a UDP flows from S to D. The other pairs of the network have a constant flow between each other (we execute the command $''pingall''$ at each host). For comparison, we ran experiments which use the Radom Route Mutation (RRM), Radom Route Mutation with Weight (RRM with Weight) and PPT-SPU respectively. The measurement results are presented in the follow.

6.2 The attacker surfaces

Fig. 5 shows the attack surfaces of the test flow over a simulation time, and the measurement adopts three approaches (RRM, RRM with Weight and PPT-SPU). In the PPT-SPU, the value of the attack surfaces has a high raise range in the start phase, then keep on a constant value. In contrast, for RRM and RRM with Weight, the attack surfaces have a low value, the raise time is brevity, the value fast grades into a constant. The strength of raising rang is determined by the introduced new path from the path set of delivering the flow. Since the segment path replacement of PPT-SPU divides a path into sub-path, the value of the attack surfaces of the two-segment replacement has a high start point and rises fast at the beginning phase. Meanwhile, benefit from a path has more than one division style, so the number of the segment combination is more than the number of the path in the path set, so the raising range of the PPT-SPU is wider than RRM and RRM with Weight. The right of **Fig. 5** is a summary of ten tests.

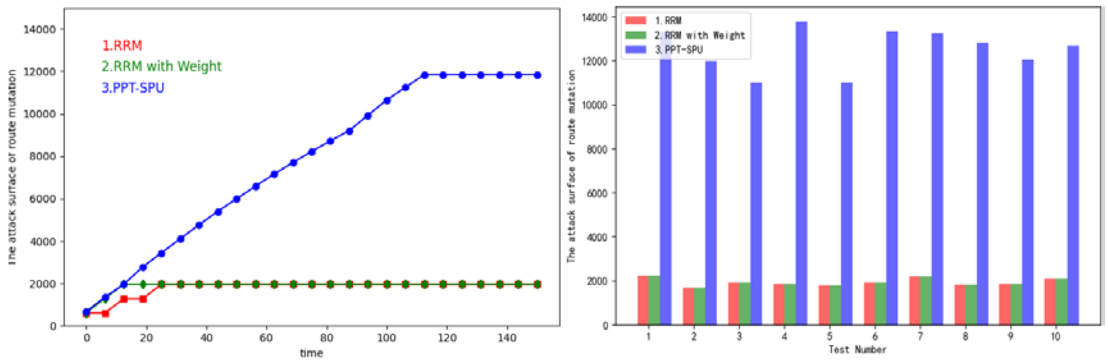

Fig. 5. The attack surfaces

6.3 The cost of control messages

Fig. 6 shows the control messaging overhead of the three approaches. For RRM and RRM with Weight, the value of the messages rises continuously and presents periodic rise. However, with PPT-SPU, the value of the messages grows linearly, and locates below the value of RRM and RRM with Weight. For adopting the RRM and RRM with Weight, a path p_i^{SD} update will bring $2 \times |p_i^{SD}|$ messages (Flow-Mod and Flow-Removed) to pave the path of pair <S,D>. In PPT-SPU, virtual segment virtual only needs 2×3 messages (Flow-Mod and Flow-Removed) for a path changing. The right of **Fig. 6** is a summary of ten tests.

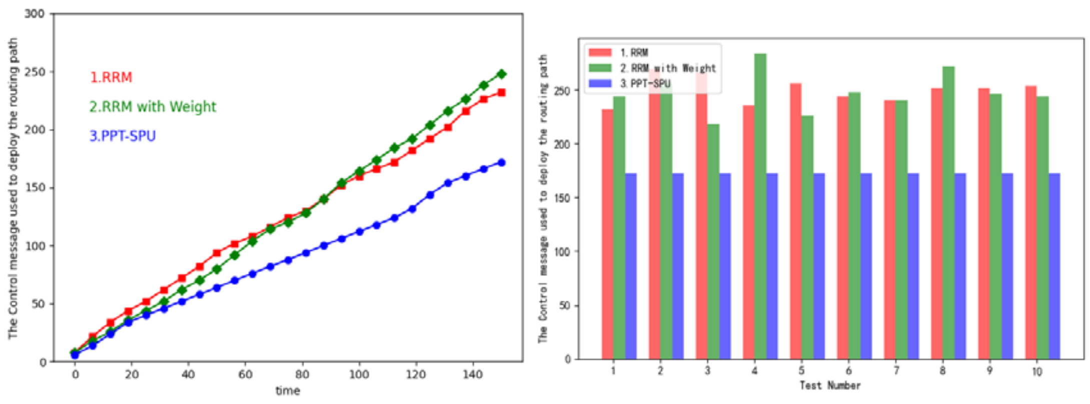

Fig. 6. Control message pay-off

6.4 The overhead on the data plane

Fig. 7 shows the storage requirement in the data plane of the three approaches. For PPT-SPU, the value is lower compared with RRM and RRM with Weight, and the value presents a periodic change. In the RRM and RRM with Weight, the values locate higher than the PPT-SPU, which due to virtual segment replacement can decrease flow entries requirement on the data plan obviously. The right of Fig. 7 is a summary of ten tests.

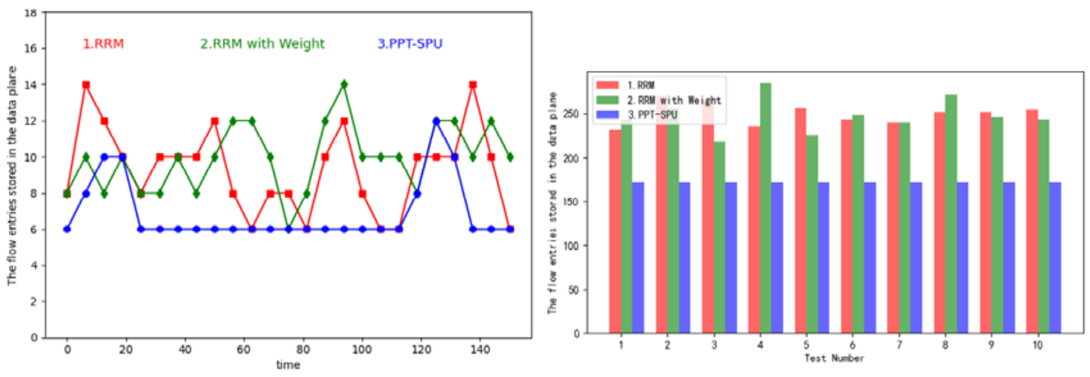

Fig. 7. Overhead on storage

6.5 Limitations

In this subsection, we will stress some limitations of our solutions. 1) Computation and storage constraints on the controller, due to introducing complex and flexible routing paradigm, PPT-SPU need more computation processes and storage on the controller. As we have known, the scalability of the central controller in SDN is an inherent problem, our method will put even more pressure on the controller. Although we believed that we can acquire enough resources above a logical center controller. However, as multi-path routing moves to a large-scale application, it is conceivable that computation and storage are needed to be considered carefully. 2) Path two-segment replacement is a typical application that can be used in route mutation. However, there is a common case that no two segments construct a path, in which needs three or more segments using to construct the path. The complex of the hopping also added. 3) PPT-SPU is firstly adopted in a telemedicine system for protecting sensitive information. Through a fog-based framework [43] attain an effective healthcare system. it still needs to verify its wide applicability in practical.

7. Conclusion

The underlay ideal of route mutation is to split a flow into a sequence of sub-flows, and these sub-flows are delivered over different paths according to the splitting sequence. The routing path mutation brings a dynamically changing attack target and enlarges the exploit surface of the attackers. However, the cost also expands to enhance security. The overhead on the control and store resources will decrease the scalability of the routing path mutation. Changing the routing path simply will bring plenty of packets that violate the regulations (e.g., shortest path first), then bring new vulnerability to the network. To cope with these problems, we proposed a PPT-SPU approach to improve the states as follows: 1) Decrease the cost of the control, and the burst caused by concurrent path changing events; 2) need a few storages on the data plane;

3) extend the attack surface; and 4) suitable to the segment routing instinctively. All in all, the proposed approach can improve the scalability of path hopping and enhance security at the same time. However, plenty of extra works are introduced into the controller at the same time, and it will bring some trouble to the logical central controller. So, more ingenious methods are needed to migrate the burden on the controller, and it is our next works also.

References

[1] German, Paul, "Time to bury dedicated hardware-based security solutions," *Network Security*, vol.8, pp.13-15, August 2017. Article (CrossRef Link)

[2] Sailik Sengupta, Ankur Chowdhary, et al., "A survey of moving target defenses for network security," *IEEE Communications Surveys & Tutorials*, vol.22, pp.1909-1942, March 2020. Article (CrossRef Link)

[3] Cho, Jin-Hee, et al., "Toward proactive, adaptive defense: A survey on moving target defense," *IEEE Communications Surveys & Tutorials*, vol.22, pp.709-745, January 2020. Article (CrossRef Link)

[4] Taguinod, Marthony, et al., "Toward a moving target defense for web applications," in *Proc. of 2015 IEEE International Conference on Information Reuse and Integration*, pp. 510-517, August 2015. Article (CrossRef Link)

[5] Jajodia, Sushil, Anup K. Ghosh, Vipin Swarup, Cliff Wang, and X. Sean Wang, eds. "Moving target defense: creating asymmetric uncertainty for cyber threats," *Springer Science & Business Media*, Vol. 54, 2011.

[6] Okhravi, Hamed, William W. Streilein, and Kevin S. Bauer. "Moving Target Techniques: Leveraging Uncertainty for CyberDefense," *MIT Lincoln Laboratory Lexington United States*, 2015. Report (CrossRef Link)

[7] Zhang, Tao, et al., "An intelligent route mutation mechanism against mixed attack based on security awareness," in *Proc. of 2019 IEEE Global Communications Conference (GLOBECOM)*, pp. 1-6, Dec. 2019. Article (CrossRef Link)

[8] Zhang, Tao, et al., "DQ-RM: Deep Reinforcement Learning-based Route Mutation Scheme for Multimedia Services," in *Proc. of 2020 International Wireless Communications and Mobile Computing (IWCMC)*, pp. 291-296, June 2020. Article (CrossRef Link)

[9] Amin, Rashid, Martin Reisslein, and Nadir Shah, "Hybrid SDN networks: A survey of existing approaches," *IEEE Communications Surveys & Tutorials*, vol.20, pp. 3259-3306, May 2018. Article (CrossRef Link)

[10] J. Vijila and A.A. Raj, "Ameliorate security by introducing security server in software defined network," *Computers, Materials & Continua*, vol. 62, no. 3, pp. 1077–1096, January 2020. Article (CrossRef Link)

[11] Zhang, Chuanhao, Youjun Bu, and Zheng Zhao, "SDN-based path hopping communication against eavesdropping attack," in *Proc. of Optical Communication, Optical Fiber Sensors, and Optical Memories for Big Data Storage*, vol.10158, pp. 101580, October 2016. Article (CrossRef Link)

[12] Zhang, Z., Deng, R., Yau, D.K., Cheng, P. and Chen, J., "On Hiddenness of Moving Target Defense against False Data Injection Attacks on Power Grid," *ACM Transactions on Cyber-Physical Systems*, vol. 4, pp. 1-29, March 2020. Article (CrossRef Link)

[13] Yang, Yubin, and Liming Cheng, "An SDN-based MTD model," *Concurrency and Computation: Practice and Experience*, vol.31, pp. e4897, October 2018. Article (CrossRef Link)

[14] B. Indira and K. Valarmathi, "A perspective of the machine learning approach for the packet classification in the software defined network," *Intelligent Automation & Soft Computing*, vol. 26, no.4, pp. 795–805, January 2020.

[15] H. Geng, J. Yao and Y. Zhang, "Single failure routing protection algorithm in the hybrid sdn network," *Computers, Materials & Continua*, vol. 64, no. 1, pp. 665–679, January 2020. Article (CrossRef Link)

[16] Jargalsaikhan Narantuya, Seunghyun Yoon, Hyuk Lim, Jin-Hee Cho, Dong Seong Kim, Terrence Moore, and Frederica Nelson, "SDN-Based IP Shuffling Moving Target Defense with Multiple SDN Controllers," in *Proc. of 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks–Supplemental Volume (DSN-S)*, pp. 15-16, August 2019. Article (CrossRef Link)

[17] Song, Fei, Yu-Tong Zhou, Yu Wang, Tian-Ming Zhao, Ilsun You, and Hong-Ke Zhang, "Smart collaborative distribution for privacy enhancement in moving target defense," *Information Sciences*, vol.479, pp. 593-606, April 2019. Article (CrossRef Link)

[18] J. H. Jafarian, E. Al-Shaer, and Q. Duan, "Openflow random host mutation: Transparent moving target defense using software defined networking," in *Proc of the First Workshop on Hot Topics in Software Defined Networks*, pp. 127–132, August 2012. Article (CrossRef Link)

[19] Qi Duan, E. Al-Shaer, and H. Jafarian, "Efficient random route mutation considering flow and network constraints," in *Proc. of IEEE Conference on Communications and Network Security (CNS)*, pp. 260–268, December 2013. Article (CrossRef Link)

[20] R. Safavi-Naini, A. Poostindouz, and V. Lisy, "Path hopping: An mtd strategy for quantum-safe communication," in *Proc. of ACM Workshop on Moving Target Defense*, pp. 111–114, October 2017. Article (CrossRef Link)

[21] Marx, Matthias, Monina Schwarz, Maximilian Blochberger, Frederik Wille, and Hannes Federrath, "Context-Aware IPv6 Address Hopping," in *Proc. of International Conference on Information and Communications Security*, pp. 539-554, February 2020. Article (CrossRef Link)

[22] Zhang, Liancheng, Yi Guo, Huiqiang Yuwen, and Yu Wang, "A port hopping based DoS mitigation scheme in SDN network," in *Proc. of 12th International Conference on Computational Intelligence and Security (CIS)*, pp. 314-317, December 2016. Article (CrossRef Link)

[23] Zhang, Liancheng, Qiang Wei, Kejun Gu, and Huiqiang Yuwen, "Path hopping based SDN network defense technology," in *Proc. of 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD)*, pp. 2058-2063, August 2016. Article (CrossRef Link)

[24] Safavi-Naini, Reihaneh, Alireza Poostindouz, and Viliam Lisy, "Path Hopping: An MTD Strategy for Long-Term Quantum-Safe Communication," *Security and Communication Networks*, vol.2018 May 2018. Article (CrossRef Link)

[25] Karim, Z. K. I. K., Anass Sebbar, Youssef Baddi, and Mohammed Boulmalf, "Secure Multipath Mutation SMPM in Moving Target Defense Based on SDN," *Procedia Computer Science*, vol.151, pp. 977-984, 2019. Article (CrossRef Link)

[26] Gillani, Fida, et al., "Agile virtualized infrastructure to proactively defend against cyber attacks," in *Proc. of 2015 IEEE Conference on Computer Communications (INFOCOM)*, pp. 729-737, May 2015. Article (CrossRef Link)

[27] Jafarian, Jafar Haadi, Ehab Al-Shaer, and Qi Duan, "Formal approach for route agility against persistent attackers," in *Proc. of European Symposium on Research in Computer Security*, Springer, Berlin, Heidelberg, pp. 237-254, Sep. 2013. Article (CrossRef Link)

[28] Liu, Jiang, Hongqi Zhang, and Zhencheng Guo, "A defense mechanism of random routing mutation in SDN," *IEICE TRANSACTIONS on Information and Systems*, 100(5), pp. 1046-1054. May 2017. Article (CrossRef Link)

[29] Duan, Qi, et al., ""Proactive routing mutation against stealthy Distributed Denial of Service attacks: metrics, modeling, and analysis," *The Journal of Defense Modeling and Simulation*, 15(2), pp. 219-230. Feb. 2018. Article (CrossRef Link)

[30] Xu, Changqiao, et al., "Context-aware Adaptive Route Mutation Scheme: A Reinforcement Learning Approach," *IEEE Internet of Things Journal*, pp.1-1 Mar 2021. Article (CrossRef Link)

[31] J. Su, R. Xu, S. Yu, B. Wang, and J. Wang, "Redundant rule detection for software-defined networking," *KSII Transactions on Internet and Information Systems*, vol. 14, no. 6, pp. 2735-2751, June 2020. Article (CrossRef Link)

[32] Deng, Wu, et al., "Differential evolution algorithm with wavelet basis function and optimal mutation strategy for complex optimization problem," *Applied Soft Computing*, 100, pp. 106724. Mar. 2021. Article (CrossRef Link)

[33] Knight, Simon, et al., "The internet topology zoo," *IEEE Journal on Selected Areas in Communications*, 29(9), pp.1765-1775. Sep. 2011. Article (CrossRef Link)

[34] Darki, Ahmad, Alexander Duff, Zhiyun Qian, Gaurav Naik, Spiros Mancoridis, and Michalis Faloutsos, "Don't trust your router: Detecting compromised router," in *Proc. of the IEEE Proceedings of the 12th International Conference on Emerging Networking Experiments and Technologies CoNEXT*, vol.16, December 2016.

[35] Liu, Huiping, Cheqing Jin, Bin Yang, and Aoying Zhou, "Finding top-k shortest paths with diversity," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, pp. 488-502, November 2017. Article (CrossRef Link)

[36] Zhang, Rongbo, Jibin Niu, Xin Li, and Shanzhi Chen, "An Anonymous System Based on Random Virtual Proxy Mutation," *Tehnički vjesnik*, vol.27, pp. 1115-1125, August 2020. Article (CrossRef Link)

[37] OpenFlow Switch Consortium, "OpenFlow Switch Specification Version 1.0.0," 2009.

[38] OpenFlow Switch Consortium, "OpenFlow Switch Specification Version 1.5.1," 2015

[39] Kollmer, Jonathan E., and Karen E. Daniels, "Betweenness centrality as predictor for forces in granular packings," *Soft matter*, vol.15, pp. 1793-1798, December 2018. Article (CrossRef Link)

[40] Keti, Faris, and Shavan Askar, "Emulation of software defined networks using Mininet in different simulation environments," in *Proc. of 2015 6th International Conference on Intelligent Systems, Modelling and Simulation*, pp. 205-210. October 2015. Article (CrossRef Link)

[41] Tu, Cheng-Chun, Joe Stringer, and Justin Pettit, "Building an extensible Open vSwitch data path," *ACM SIGOPS Operating Systems Review*, vol. 51, pp. 72-77, September 2017. Article (CrossRef Link)

[42] R. C. Meena, M. Bundele and M. Nawal, "RYU SDN Controller Testbed for Performance Testing of Source Address Validation Techniques," in *Proc. of 2020 3rd International Conference on Emerging Technologies in Computer Engineering: Machine Learning and Internet of Things (ICETCE)*, pp. 1-6, February 2020. Article (CrossRef Link)

[43] L. Sun, Q. Yu, D. Peng, S. Subramani and X. Wang, "Fogmed: a fog-based framework for disease prognosis based medical sensor data streams," *Computers, Materials & Continua*, vol. 66, no.1, pp. 603–619, October 2020. Article (CrossRef Link)

Rongbo Zhang is currently pursuing the Ph.D. in Beijing University of Posts and Telecommunications (BUPT), China. He received his Master of Science in Engineering from Information Engineering University and Bachelor degree from Zhengzhou University (Zhengzhou, China). He is a teacher of Zhengzhou University (Zhengzhou, China). He currently participates in the research and development work related to SDN.

Xin Li received his Ph.D. degree in Communication and Information Systems from BUPT, China, in 2007. Then he joined BUPT and served as associate professor working in State Key Laboratory of networking and switching technology. His current research interests include software-defined networking, content distribution network, Big data information and theory, network reliability and survivability.

Zhan Yan is currently a senior project engineer of AVI-SPL. He became a member of AVE-TELAV as a system design specialist in 2011. He joined sharp's audio visual in 2012. His research interests include audio and video equipment design, information security, computer networks reliability, and survivability.