MASK: Practical Source and Path Verification based on Multi-AS-Key

Songtao Fu*,Ke Xu*†§,Qi Li†‡, Xiaoliang Wang¶, Su Yao†, Yangfei Guo‡ and Xinle Du*

* Department of Computer Science and Technology, Tsinghua University, Beijing, China

† Beijing National Research Center for Information Science and Technology(BNRist), Beijing, China

‡ Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing, China

§ Peng Cheng Laboratory, Shenzhen, China ¶Capital Normal University, Beijing, China

{fust18@mails.tsinghua.edu.cn, xuke@tsinghua.edu.cn, qli01@tsinghua.edu.cn, wangxiaoliang@cnu.edu.cn, yaosu@tsinghua.edu.cn, guoyangf19@mails.tsinghua.edu.cn, dxl18@mails.tsinghua.edu.cn}

***Abstract*—The source and path verification in path-aware Internet consider the two critical issues: (1) end hosts could verify that their forwarding decisions followed by the network, (2) both intermediate routers and destination host could authenticate the source of packets and filter the malicious traffic. Unfortunately, the current verification mechanism requires validation operations in each router on the path in an inter-domain environment, thus requiring high communication and computation overhead, reducing its usefulness; besides, it is also difficult to meet the dynamic requirements of the end host. Ideally, the verification should be secure and provide the customized capability to meet the end host's requirements. We propose a new mechanism called source and path verification based on Multi-AS-Key (MASK). Instead of each packet verified and marked at each router on the path, MASK improves the verification by empowering the end hosts to instruct the routers to achieve the verification, thus decreasing the router's overhead while ensuring security performance to meet the end host's requirements. With the plausible design, the communication overhead for realistic path lengths is 3–8 times smaller than the state-of-the-art mechanisms. The computation overhead in the routers is 2-5 times smaller. We implement our design in the BMv2 environment and commodity Barefoot Tofino programmable switch, demonstrating that MASK introduces significantly less overhead than the existing mechanisms.**

***Index Terms*—path-aware networking, data plane, source and path verification**

I. INTRODUCTION

Security threats and attacks on Internet have been increasing at an alarming rate in recent years. The untrusted network layer leads to many problems, such as the TCP hijack by manipulating the IPID assignment method [1], or Certificate Authorities (CAs) are vulnerable to attacks by network layer adversaries [2]. A secure data plane is needed on Internet [3]. With the exciting research of path-aware networking architectures on the Internet, the end hosts could impose their policies to enforce the forwarding decisions on the data plane [4] [5] [6]. It provides an opportunity for establishing a secure data plane in the network layer.

The source and path verification in path-aware networking could fundamentally improve the network's trustworthiness, which achieves two fundamental security properties: (1) The end hosts (the source and the destination) could verify that the network follows their forwarding decisions. (2) The intermediate routers and destination hosts could authenticate the packets' source while filtering the malicious traffic as early as possible. To achieve the verification, the source, intermediate routers as well as destination should calculate marks with the session keys shared between these entities [7]. The granularity of verification could be AS level since that an AS is separately administered as a trust and fate-sharing unit [8]. Some sophisticated mechanisms devote to improve efficiency and security, but the following deficiencies still degrade the practicality of these mechanisms:

High communication overhead. The state-of-the-art additional packet header still impractical in the inter-domain communication [7] [9], while the existing probabilistic mechanism [10] devotes to decrease the communication overhead also degrades the security performance, e.g., lack of the ability to filter the malicious traffic in routers.

Heavy computation overhead. Sharing the per-session keys between the end hosts and routers requires sophisticated actions beyond the capability of today's routers, e.g., the asymmetric encryption is impractical at least with today's router hardware, the mechanism such as DRKey [7] couldn't be used directly. EPIC decreases this overhead with a practical hierarchical key derivation mechanism [11]. But even symmetric cryptographic operations in the data plane could decrease the line rate performance, e.g., the cryptographic operation implemented by the software could be a bottleneck of the forwarding, while utilizing the dedicated hardware increases the cost as well as the difficulty to upgrade the relative routers.

Impractical time synchronization. Some mechanisms utilize the timestamp to resist the attacks, such as replay attack, which needs synchronization between the routers and end hosts [12] [13]. It is also impractical to keep the precise data plane synchronization between these entities for today's inter-domain communication.

This paper proposes MASK, a practical source and path verification with an efficient additional packet header and lightweight operation in the router. We offload both of the overhead to the end host and the Key Distribution Server (KDS). The KDSes in different ASes share Multi-AS-Key with the state machine in APPA [14], to derive the session keys for the hosts and intermediate routers, then distribute them to the end hosts as well as routers in the same AS in a confidential way. A source host could get the path information and the session keys from the KDS. Then it could negotiate a per-session policy with the destination and instruct the routers to perform a lightweight operation according to the policy. For a certain packet, only one hop[1] on the path performs one symmetric cryptographic operation, while the source and the destination could validate the processing according to the policy. The main contribution of this paper is as follow:

- Empowering the end host to instruct the intermediate routers to achieve the verification with the policy negotiated by the source and destination.
- Introducing a lightweight mechanism to operate source and path verification in intermediate routers.
- Evaluating the performance and comparing it with the state-of-the-art design in the BMv2 environment and Barefoot Tofino programmable switch.

The rest of the paper organizes as follows: In the next section, we analyze our design's background and motivation. Section III presents the MASK design at a high level, section IV details the MASK protocol, section V analyzes its characteristics and benefits, section VI presents the implementation as well as the evaluation, and section VII discusses the incremental deployment mechanism as well as the limitation. Section VIII describes related works. We conclude the paper in the last section.

II. Background and Motivation

This section presents the background of source and path verification on the Internet, analyzes the challenges in a realistic network, and then explains our design goals.

A. Background and Challenges

As the source and path verification focus on securing the data plane, consider a Dolev–Yao adversary could observe, drop, inject, replay, or alter packets in a compromised network [15]. The intermediate routers and end hosts must share some secret information (session keys) with each other [16], to generate cryptographic marks for each hop in the packet header for cryptographically verifying the source and path information during forwarding [7] [9]. The state-of-the-art cryptographic mechanisms include the FULL and probabilistic process. The destination achieves the verification with the marks in both cases. The FULL operation (includes OPT [7], EPIC [9], et.al.), as shown in Fig. 1 (a), verifies the $mark_{src}$ and adds its $mark_r$ at each hop. In contrast, as shown in Fig. 1 (b), the probabilistic process (PPV [10]) adds a specific number of $mark_r$ (e.g., two hops). The former couldn't utilize the source's (S) and destination's (D) ability to verify a specific session. The latter cannot filter the malicious traffic based on the source's $mark_{src}$.

In a word, a practical verification mechanism in a realistic network should satisfy the requirement of scalability and

[1]We denote hop as an AS-level border router in this paper.

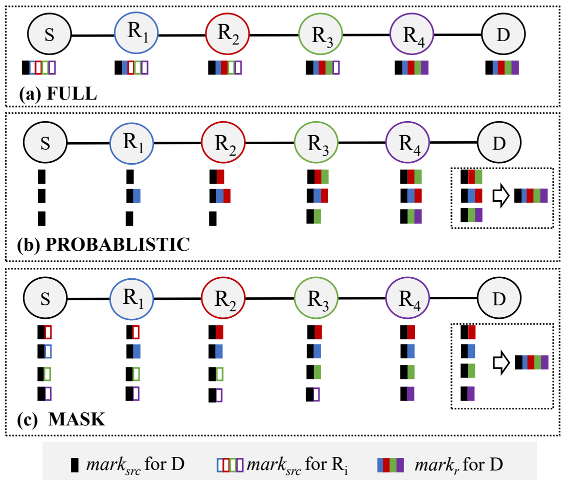


Fig. 1. The overview of different verification mechanisms

deployability. Today's designs are lacking in both regards, which brings the challenges as follows:

Scalability. The verification mechanism shouldn't cripple the performance of today's Internet. A lightweight computation overhead in routers with hundreds of nanoseconds is necessary for inter-domain communication [9]. Meanwhile, the significant overhead in routers not only degrades the performance but also gives the attackers the chance to simply flood a router with illegitimate packets [17].

Deployability. The Internet is a network of networks connecting billions of hosts in more than 60,000 AS [18]. Hardware routers used to connect the ASes are still extremely hard to upgrade; even adding simple functionality is difficult to achieve without vendor support. Meanwhile, legacy routers need to communicate with updated routers and vice versa.

From this perspective, we believe that a lightweight router could offload the overhead to other entities, e.g., the end host. As shown in Fig. 1 (c), a router could achieve the verification with the source's instruction to minimize the router's overhead. But how can MASK maintain the security guarantees on the premise of the lightweight router?

B. Design Goals

The design should be implemented on routers within reach of the present or foreseeable future technology. We begin with the following assumptions:

- Secure Intra-AS Communications. We assume a secure channel to communicate between a host and KDS in its own AS to get the session keys for the routers and hosts on the forwarding path. In particular, an AS may prevent source address spoofing within its network using any anti-spoofing method such as SAVA [19] [20] [21].
- Hierarchical Key Establishment. We leverage the KDSes to establish the 3 levels key system. As shown in Table [I], The AS-Key (AK) is the top-level key infrequently renewed (e.g., hours) with the state machine in APPA [14], only shared among the KDSes with hash chain [22].

At the same time, the router key (RK) and the session key (RSK, HSK) is the level 2 and 3 key. The source and destination could get the RSK and HSK from the relative KDS; the router can dynamically recreate RSK on the fly to ensure the confidentiality of RK while needn't keep a per-session state.

TABLE I
HIERARCHICAL KEY DERIVED BY KDS

Notation	Description	Level	Storage
AK	AS-Key	1	KDS
RK	Router Key	2	KDS, router
RSK	Router's Session Key	3	KDS, host
HSK	Host's Session Key	3	KDS, host

As the above assumptions have already been deployed in the realistic network, based on these assumptions, the router could offload its overhead to the end host and the KDS. Then we could achieve the following design goals:

Scalability. We aim to design an edge-based solution to empower the distributed end host by imposing its policy on the network. This end host driving mechanism minimizes the cryptographic operations in the routers while keeping the security characteristic. For a packet, only one hop on the path needs to execute a cryptographic operation to verify the source's mark and add a new mark to be verified by the destination.

Deployability. MASK should not be difficult to be deployed or upgraded in different environments (e.g., an inter-domain or dedicated network). The emerging programmable switch [23], which supports packet processing using domain specific languages (e.g., P4 [24]), provides an exciting opportunity to design protocols that manipulate arbitrary parts of packet header at line rate [25]. It still has some challenges to utilize the programmable switch since the ASIC constraints though some new extension of P4 has been put forward [26]. However, MASK's design is devoted to addressing these challenges by minimizing the verification's operation.

III. OVERVIEW

This section overviews the design of MASK. We will first detail the additional packet header in MASK and then show how MASK processes at the end host and router.

To achieve the verification, we design the packet header with IPv6 Extension Header (EH) [27], as shown in Fig. 2, the additional packet header of MASK are as follow:

- $Flag$ (8-bits): We denote the two least significant bits of $Flag$ with the value 0, 1, 2, 3 to indicate the packet of DATA, NegoPkt, ACK, CHECK, respectively.
- ID (32-bits): Identifier of an intermediate router.
- $session_{partial}$ (32-bits): The substring of the partial session information calculated as Equation (1), in which ID_i is the ID of an intermediate router, TS is the initial time of a session with the granularity of second, '$||$' represents the concatenation, $H(.)$ represents a certain

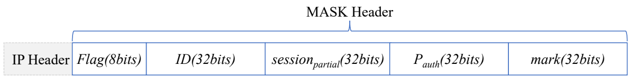


Fig. 2. The packet header of MASK

hash operation, while [0:32] represents truncating the lowest 32 bits field. A session could be identified by the concatenation of $session_{partial}$, IP_{src} and IP_{dst} (the IP address of the source and the destination).

$$\begin{aligned} session_{partial} &= H(ID_1||\ldots||ID_n||TS)[0:32] \\ session &= IP_{src}||IP_{dst}||session_{partial} \end{aligned} \tag{1}$$

- P_{auth} (32-bits): The authenticator calculated by the source as Equation (2), in which $MAC(.)$ represents the MAC computation, $hash$ is the hash of packet payload, to guarantee the diversity of the marks while the destination could verify the source and the integrity of the packet.

$$P_{auth} = MAC_{HSK}(hash)[0:32] \tag{2}$$

- $mark$ (32-bits): The authenticator calculated by the source as well as the router.

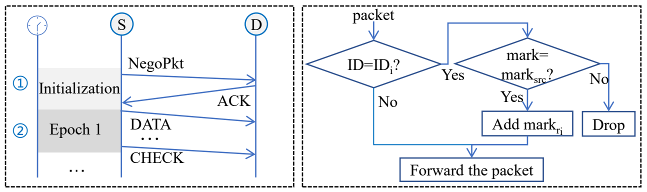
At a high level, MASK offloads most of the router's overhead to the end host. Specifically, the router could probabilistically verify and mark a packet. The end host could achieve a dynamic verification with a negotiating policy.

Processing at end host. As shown in Fig. 3 (a), the source (S) and destination (D) could require the path information and the session keys from the KDS in the relative AS, then the processing consists of two phases: ① Initialization: the source sends a negotiation packet (NegoPkt) includes the path information as well as the policy towards the destination, nests an intermediate router's ID_i as well as a $mark_{src_i}$ in the packet header, the destination verifies the packet and returns the ACK (the NegoPkt and ACK can carry upper layer protocol data). ② Verification in each epoch: The source selects an intermediate router according to the policy, nests its ID_i as well as a $mark_{src_i}$ in the DATA packet (DATA) header, the destination achieves the verification after receiving each packet. At the end of each epoch, the source sends a checkpoint packet (CHECK), the destination validates the processing according to negotiating policy after successfully receive a CHECK.

Processing at router: As shown in Fig. 3 (b), for a packet, a router with the same ID_i verifies the $mark_{src_i}$ and replaces it with $mark_{r_i}$ after a successful source verification, or else drops the packet which failed of source verification. The routers without the same ID_i forward packets directly.

IV. PROTOCOL DESCRIPTION

In the following subsections, we first introduce the dynamic processing includes the instruction at the source and the verification at the destination, then explain the lightweight processing at the router. After that, we describe the key switching mechanisms to guarantee communication and then talk about path stability effects.



(a) Processing at end host (b) Processing at router

Fig. 3. The MASK overview

A. Processing at End Host

The source first requires the path information as well as the relative session keys from its KDS. Given an AK in that interval, the KDS could derive the level 2 key, which is the RK_i has sent to the relative router as Equation (3), and the level 3 key includes the RSK_i as Equation (4), as well as the HSK as Equation (5).

$$RK_i = MAC_{AK_{ID_i}}(ID_i) \tag{3}$$

$$RSK_i = MAC_{RK_i}(session) \tag{4}$$

$$HSK = MAC_{AK_{src}}(session) \tag{5}$$

The source gets the path information and session keys of routers as Equation (6), as well as the HSK, then sends a NegoPkt towards the destination, with the policy encrypted by the HSK. We assume a simple policy with a Checkpoint Sequence Hopping (CSH) message, e.g., the CSH of '4000|5000 $\cdots$' represents the source sends 4000 packets in the first epoch and 5000 packets in the second epoch. The destination gets the keys from the KDS in destination AS, verifies the packet and decrypts the policy, changes the CSH to $CSH' = CSH||IP_{dst}$, encrypts the CSH' with the HSK, and sends an ACK towards the source. After the source decrypts the CSH', the end hosts establish the initialization.

$$\begin{aligned} path &= (ID_1, ID_2, \cdots, ID_n) \\ RSK_{path} &= (RSK_1, RSK_2, \cdots, RSK_n, HSK) \end{aligned} \tag{6}$$

After the negotiation, with a certain policy (e.g., routers marks with the uniform probability), in each epoch, the source forwards the packet as Algorithm 1.

In Algorithm 1, the source already has the $path$, RSK_{path}, HSK and the CSH, selects $CSH[m]$ from CSH (e.g., 4000) in each epoch (line 1). Then it selects one router for each packet according to the policy (e.g., uniform marking) and nests the $mark_{src_i}$ as Equation (8) (line 3-5).

$$mark = MAC_{RSK_i}(P_{auth}||ID_{i-1}||ID_{i+1}) \tag{7}$$

$$mark_{src_i} = mark[0:32] \tag{8}$$

If the $CSH[m]$ decreases to 0, the source sends a CHECK (line 6-8). A CHECK is different from the DATA, the P_{auth} was replaced by P^{csh}_{auth} calculated with the policy in the next epoch, while the $mark_{src_i}$ is the same as a DATA.

$$P^{csh}_{auth} = MAC_{HSK}(CSH[m+1])[0:32] \tag{9}$$

Algorithm 1: procedure at source host

```
     input : $path, RSK_{path}, HSK, CSH$
 1   for $CSH[m]$ in $CSH$ do
 2       while $CSH[m] >= 0$ do
 3           if $CSH[m] > 0$ then
 4               $flag \leftarrow 0, ID \leftarrow ID_i$
 5               $P_{auth} \leftarrow P_{auth}, mark \leftarrow mark_{src_i}$
 6           else
 7               $flag \leftarrow 3, ID \leftarrow ID_i$
 8               $P_{auth} \leftarrow P^{csh}_{auth}, mark \leftarrow mark_{src_i}$
 9           $CSH[m] \leftarrow CSH[m] - 1$
10           send packet to the first hop
```

Then the source decreases the $CSH[m]$ and sends the packet to the first hop (line 9-10). It could begin a new turn with $CSH[0]$ after all the CSH was over or negotiate a new policy with the destination.

The destination achieves the source and path verification with Algorithm 2.

Algorithm 2: procedure at destination host

```
     input : $pkt, path, HSK, RSK_{path}, CSH$
 1   initialize $sCouter[N] = \{0\}, fCounter[N] = \{0\}$
     if ($flag == 0$) and ($P_{auth} == P'_{auth}$) then
 2       if $mark_{r_i} == mark'_{r_i}$ then
 3           $sCouter[i]++$
 4       else
 5           $fCounter[i]++$
 6   else if ($flag == 3$) and ($P^{csh}_{auth} == {P^{csh}_{auth}}'$) then
 7       if $mark_{r_i} == mark'_{r_i}$ then
 8           for $sCouter[i], fCounter[i]$ in
               $sCouter[N], fCounter[N]$ do
 9               if all the $sCounter[i]$,$fCounter[i]$
                   match the policy then
10                   m= (m+1) MOD length (CSH)
11                   $sCouter[N], fCounter[N] = \{0\}$
12               else
13                   fault router $i$ is located
```

The destination initializes two counter, the $sCouter[N]$ represents the counter the relative router successfully pass the verification, while the $fCounter[N]$ represents the counter failed the verification (line 1). If the packet is a DATA, the destination calculates the P'_{auth} and compares it with P_{auth} in the packet, then calculates the $mark'_{r_i}$, compares with the $mark_{r_i}$ in the packet header (line 2-3). If the verification is successful, the destination increases the relative router's $sCouter[i]$, or else increases the $fCounter[i]$ (line 4-5). At the end of each epoch, the destination receives the CHECK, it first verifies the P^{csh}_{auth} in the packet (line 6), then calculates the $mark_{r_i}{}'$ and compares with $mark_{r_i}$ (line 7). Whether the $sCouter[i]$ and $fCounter[i]$ match the policy could be judged as follow:

$$\alpha * CSH[m] * (1-\delta) < sCouter[i]) < \alpha * CSH[m] * (1+\delta) \quad (10)$$

$$fCounter[i] < 2 * \delta * CSH[m] \quad (11)$$

In Equation (10) α is the marking probability of each router, e.g., for a uniform mechanism with 5 hops, $\alpha = 0.2$, δ is the threshold according to the statistical network performance (e.g. 0.05 for a natural loss of 0.03 [28]). While in Equation (11) $2*\delta*CSH[m]$ is the upper threshold of the failed verification counter. If the two counters comply with the policy, the destination turns into the next epoch and resets the counters (line 8-11). Otherwise, the destination localizes the router i as a fault node (line 13). With this design, the destination could detect the replay attack since it violent the policy. The destination could report the verification encrypted with the HSK towards the source. We do not assume an exact report mechanism towards the source or other entities, as the reply could send with the application layer information. It worth mentioning that the ACK and the DATA from the destination to the source on the reverse path are similar to the forward path.

B. Processing at Router

When a router receives a packet, it processes the packet as Algorithm 3. It first checks whether it's a packet of MASK, and checks that whether the ID_i is the same as itself (line 1), calculates the $mark'_{src_i}$ with the router's session key RSK_i as Equation (8), then compares $mark'_{src_i}$ with the $mark_{src_i}$ in the packet header. The router would drop the packet if the source verification failed, to filter the malicious traffic (line 2-3). If the $mark_{src_i}$ passes the source verification, the router replaces the $mark_{src_i}$ with $mark_{r_i}$ as Equation (12) (line 4-5), then the router sends it to the next hop (line 6).

$$mark_{r_i} = mark[32:64] \quad (12)$$

With this design, the router only performs one MAC operation, truncated it for the source and path verification.

```
Algorithm 3: procedure at router
   input : $pkt, RK_i, ID_i, ID_{i-1}, ID_{i+1}$
1  if ($flag == MASK$) and ($ID_i == ID_i{}^{pkt}$) then
2      if $mark_{src_i} != mark'_{src_i}$ then
3          drop the packet
4      else
5          $mark_{src_i} \leftarrow mark_{r_i}$
6  send packet to the next hop
```

C. Key Switching

The key switching is infrequent since the AK changes with the granularity of hours. However, a key switching mechanism is still needed to prevent the session from interrupting since letting all entities change keys simultaneously is impossible. We introduce offset time to deal with the switching. Each entity prepares the new key before the offset time. During the offset time, the source sets the third least significant bit in $Flag$ with 1 to announce the entities to use the new key (all the entities use the old key with 0). After the offset time, all the entities utilize the new key and drop the old key no matter the $Flag$ value. Keeping two keys is easy to be accomplished by the destination host while incurring little overhead in the router. It's better than invalidate an entire session and force the source and destination to execute another handshake simply because a few entities switch their keys earlier or later than others.

D. Path Stability Effects

Localized load-balancing does not hurt MASK's performance since MASK works in AS granularity. The large-scale route flapping is detrimental to communication performance. It is orthogonal to MASK since the routing decisions should avoid the route flapping based on the observation that most networks are nearly quiescent (in terms of routing changes) [29]. Suppose a path changes during the transmission for some reasons (e.g., infrastructure failures or temporary outage). In that case, the relative session keys will change, it will force the source to renegotiate before sending further marked packets. An alternative that may mitigate the effect of mid-flow route changes is that the source can simply demote the packet to unmarked status during the renegotiation.

V. ANALYSIS

In this section, we analyze the performance as well as security properties of MASK. We first compare the characteristics with the state-of-the-art mechanisms, then analyze how to deal with various attacks.

A. Performance Analysis

Overhead. As shown in Table [II], we compare the computation and communication overhead with OPT [7], EPIC L3 [9] and PPV [10] since they provide similar security guarantees.

- Computation overhead. The cryptographic operation accounts for most of the computation overhead. MASK efficiently optimizes the per-packet cryptographic operation with only one hop on the path to operate the MAC calculation. While OPT and EPIC L3 operate at each hop, the PPV operates at two adjacent hops.
- Communication overhead. The MASK and PPV additional packet header size is stable, while the OPT and EPIC L3 header size depends on the path length. We illustrate the communication overhead with 5 and 10 hops since the current average path length on the Internet is less than 5 AS-level hops, and the vast majority of Internet AS-level path length is at most 10 [30].

TABLE II
THE OVERHEAD COMPARED WITH OPT, PPV AND EPIC

	Computation overhead (hop)	Communication overhead		
		size (B)	for n=5 (B)	for n=10 (B)
OPT	n	16n+68	148	228
PPV	2	64	64	64
EPIC	n	5n+24	49	74
MASK	1	17	17	17

Convergency delay. As we utilize the probabilistic mark mechanism, the destination will reconstruct the actual forwarding path after receiving a certain number of packets. We denote it as convergence delay T_c which indicates the expected number of received packets used to reconstruct the actual path in the destination. While T_c is proportional to time for a certain rate. A useful link defined in MASK is a link not appearing in the previous packets of the session. As $P_0 = 1/n$, the probability of receiving a useful link is nP_0 and $(n-1)P_0$ respectively from the first and second received packet, and $(n-j+1)P_0$ from the $j-th$ received. As detailed in [10], T_c could be calculated as Equation (13), in which $\omega = 0.577$.

$$\begin{aligned} T_c &= \frac{1}{n*P_0} + \cdots + \frac{1}{(n-j+1)*P_0} + \cdots + \frac{1}{P_0} \\ &\approx n(lnn + \omega) \end{aligned} \tag{13}$$

B. Security Analysis

Security properties. An overview of the comparison with OPT [7], PPV [10] and EPIC L3 [9] is shown in Table [III]. No asymmetric cryptography represents that the router needn't perform the per-session asymmetric cryptography. EPIC and MASK could derive the session keys from the key server; others have to negotiate the session keys by asymmetric cryptography. The router's source and path verification is the honest router's capability to filter the traffic from a malicious source and an on-path attacker. PPV couldn't achieve this since the routers only add its own mark, while in MASK, the source decides which router to perform the verification. The source and path verification in the destination ensures that the destination could authenticate the source and verifies whether the actual path corresponds to the anticipated path. EPIC achieves replay attack resistance with the timestamp, but it's impractical for the entities to keep time synchronization in a large scale network. We relax it with the policy in the granularity of an epoch, and the destination could achieve it at each end of an epoch. Meanwhile, a malicious router might drop packets during the processing; only MASK could detect the packet drop attack according to the policy, while others couldn't achieve that for that they only achieve the verification according to received packets.

Packet flooding. In MASK, an unauthorized entity might forge the appropriate mark by packet flooding. The shorter length and the less diversity of the mark provide the larger probability that a randomly chosen mark will pass through the verification. We have the parameters: z, the length of a mark, and x, the size of P_{auth} represents the diversity of marks. The probability that a randomly guessed mark equal to the particular router is as:

$$P(x,z) = \frac{1}{2^{min(x-1,z-1)}} \tag{14}$$

In which $min(a,b)$ represents the minimum value between a and b would be selected. MASK has 32 bits x and 32 bits z. It's enough to achieve verification requirements [28].

Attackers could also simply flood a router with illegitimate marks, causing the router to either overload its computation capability or fill its buffer with packets. MASK decreases the overload risk since only one hop on the path operates the verification. An attacker could also send packets with the ID not in the forwarding path to evade the verification, which's equal to flood with unmarked packets. MASK could efficiently deal with this case by the design that the MASK packet has higher priority. At the same time, the destination could easily filter the packet with the wrong ID. In a word, packet flooding couldn't degrade or evade the verification.

Coward attack. An attacker may mount a coward attack [31] to evade detection after observing the location or the pattern of ID. To prevent an attacker from observing the location of ID, we could translate the ID to a dynamic ID in the source calculated as:

$$ID_i{}' = MAC_{RSK_i}(P_{auth}||ID_i)[0:32] \tag{15}$$

With this design, a router couldn't know which router should verify and mark the packet, except it is the node that needs to mark the packet.

To prevent an attacker from observing the pattern of ID, the source and the destination could generate a more sophisticated policy. For example, the source announces a sequence just like '00101011...', in which '0010', '1011'... represent a router's ordinal number on the path (assuming the path length is less than 16). Then the source selects the router according to this sequence repeatedly until the end of each epoch. The sequence could generate by the session key of the two end hosts with a pseudorandom function.

Moreover, attackers might compromise the verification system from non-deployed domains. In this case, MASK should verify the non-deployed source's scheduled path in the ingress border router of the deployed domain with the $mark$ in the packet header. A malicious end host might launch a man-in-the-middle attack. MASK could guarantee security by the secure intra-AS communication or the access authentication with identifier [32] [33] [34]. An illegal entity can't obtain the correct session key, thus has no opportunity to evade the verification.

VI. EVALUATION

To show that MASK is practically feasible, we evaluated MASK on both software (Bmv2[2]) and hardware (Barefoot Tofino programmable switch). The following evaluation shows

[2]https://github.com/p4lang/behavioral-model

TABLE III
THE COMPARISON OF SECURITY PROPERTIES WITH OPT, PPV AND EPIC

	Router			Destination			
	No asymmetric cryptography	Source verification	Path verification	Source verification	Path verification	Replay attack resistance	Packet drop
OPT	×	✓	✓	✓	✓	×	×
PPV	×	×	×	✓	✓	×	×
EPIC L3	✓	✓	✓	✓	✓	✓	×
MASK	✓	✓	✓	✓	✓	✓	✓

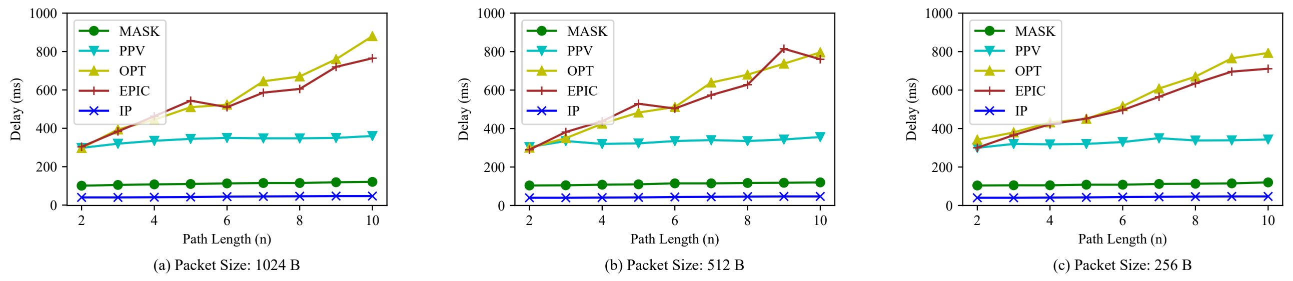


Fig. 4. The end-to-end delay in BMv2 environment

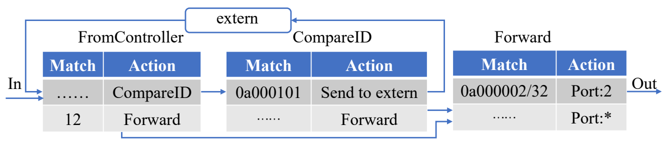


Fig. 5. The processing of MASK in BMv2

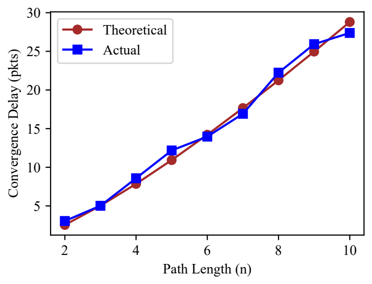


Fig. 6. The convergence delay in different path length

that MASK has significantly lower computation and communication overhead.

A. Evaluation in BMv2

We implemented MASK in the BMv2 environment, the testbed hosts in a virtual machine with Ubuntu 16.04, with the Intel Core i5-6200U CPU, 2.4 GHz, and 4 GB RAM. We instantiate MAC operation with the HMAC based on MD5 and truncate the $mark$ to 32 bits.

Fig. 5 shows the processing of MASK. There are three Match-Action tables in the ingress pipeline. The first ($FromController$) is to judge whether the extern (MAC operation) has processed the packet and sends a processed packet to the third table ($Forward$) for forwarding. In contrast, it sends an unprocessed packet to the second table ($CompareID$). The second is for comparing the ID_i, which decides whether sends the packet to extern or forwards the packet with the third table directly. The extern calculates the $mark'_{src_i}$ and achieves the source verification, then filters the malicious traffic or adds its $mark_{r_i}$ after successful verification.

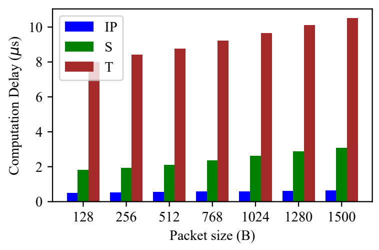
As shown in Fig. 4 (a),(b),(c), we evaluate MASK in terms of end-to-end delay, which composes of the computation and forwarding delay. The baseline is the IPv6 packet with nearly 50 ms in different path lengths and packet sizes. MASK and PPV have the constant delay of 110 ms and 300 ms respectively (PPV needs to calculate an additional Marking Verification Field). In contrast, OPT and EPIC have a relatively high delay since they operate the MAC operation at each hop.

When it comes to the convergence delay, we calculate the theoretical delay in different path lengths with Equation (13). As shown in Fig. 6, the actual delay (the average packets that the destination reconstructs the path) is consistent with the theoretical delay with the value around $2n$ for most situations.

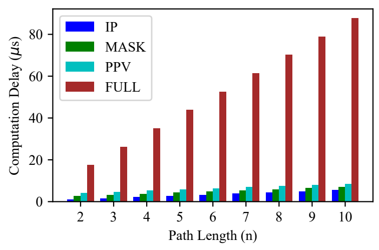
B. Evaluation in Barefoot Tofino Programmable Switch

We evaluate MASK for the desired performance in commodity Barefoot Tofino programmable switches S9180-32X.

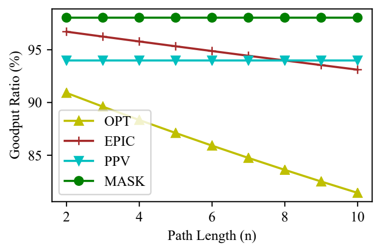
We first examine the computation overhead in terms of per-packet computation delay. The computation delay on a certain hop includes (1) source verification (calculating and verifying the source's mark), (2) path verification (calculating the router's mark). We implemented the MAC operation as AES operation [35]. MASK has a significantly lower delay than other mechanisms.



(a) The computation overhead in different packet size



(b) The computation overhead in different path length



(c) The goodput ratio in different path length

Fig. 7. The computation overhead and goodput ratio in Barefoot hardware

The delay in different packet sizes is shown in Fig. 7 (a), in which IP represents IPv6 packet forwarding, which is the baseline with the value of around 0.55 μs, S represents single AES operation, while T represents two AES operation. We found that a single AES operation with the AES-128 algorithm (10 rounds) is nearly 2.1 μs with the packet size of 512 Bytes. In contrast, two AES operations take nearly 8.8 μs. The results confirm that the most significant delay comes from the AES operation, and the delay grows with the packet size since a packet recirculates several times to complete one AES operation.

We compared the FULL operation (OPT and EPIC) and PPV operation with MASK in path length from 2 to 10 with the packet size of 512 Bytes. As shown in Fig. 7 (b), MASK optimizes the operation. In any case, only one hop verified and added the mark; in PPV, two adjacent hops perform one AES operation. In contrast, the FULL operation has more latency with a longer path. It worth mentioning that a dedicated cryptography circuit could improve the cryptography performance, e.g., Intel's AES-NI hardware [9], but it could also incur complexity as well as the difficulty to upgrade.

A central aspect of our work is throughput. MASK could achieve higher throughput than other mechanisms. The throughput was affected by the hardware's recirculation bottleneck since a packet recirculates several times to complete one AES operation, the highest throughput achieves 10.92 Gbps in [35]. We utilize a 10Gbps Intel 82599ES NIC to connect with the switch hardware. The switch receives the packet from NIC, deals with the verification, and then forwards back to the NIC with the highest throughput of 8.02 Gbps. Simultaneously, the baseline throughput without AES operation is 8.84 Gbps in the same hardware, which testifies that the FULL operation's highest throughput (e.g., OPT, EPIC) is 8.02 Gbps. In contrast, with the probabilistic design in MASK, only 1/n packets need to recirculate in a hop. The throughput could achieve 40 Gbps with a path length of 5. Consider the user can configure more physical ports to increase throughput, we can conclude that the throughput is sufficient for supporting the verification.

When it comes to the goodput, we use the IPv6 extension header to load the security-related header fields and define the additional packet header size as HD. The goodput ratio (GR) is the ratio of payload size (p) and total packet size (p+HD), namely GR = p/(p+HD). As shown in Fig. 7 (c), the goodput ratio is high for MASK. For the payload size of 1000 Bytes in n = 5, the additional header in EPIC L3 is 49 Bytes, which corresponds to a goodput ratio of 95.3%, while the value in OPT is 87.1%. In MASK, the ratio is stable at 98.3% regardless of the path length, while PPV has a stable ratio with a value of 94.0%.

C. Overhead in End Host

In MASK, the end host requests path and session keys from the KDS, the KDS answers with the path and relative session keys before a new session. We assume that the underlying path-aware networking minimizes latency by locally caching public paths. And the Multi-AS-Key has been synchronized among the KDSes. The additional latency incurred in MASK is insignificant since the paths and keys information is available at local KDS. A host can cache both paths and keys, which eliminates extra latency for subsequent packets.

In comparison, the negotiating process could combine with TCP handshake and then decrease the extra latency. As the source and destination need to perform a MAC operation for each packet, which comprises the heaviest overhead, we test AES-128 operation with 10 rounds. It could achieve more than 150,000 cycles per second in the virtual machine with Intel Core i5-6200U CPU, which indicates that it could support nearly 1 Gbps with a packet size of more than 500 Bytes in an ordinary commodity host.

VII. DISCUSSION

In this section, we discuss incremental deployment issues and the limitations of MASK. We first discuss a high-level approach for implementing MASK on the Internet, then discuss attacks that MASK does not entirely defend.

A. Incremental Deployment

We emphasize implementing with IPv6 extension header to enjoy compatibility. The IPv4 ecosystem could also use MASK. As the Internet's architecture is a flexible composition of many networks [36], a dedicated network could use some new designs. From this perspective, we insist that MASK could first deploy in some dedicated network, just like the verifiable service function chains (SFCs) [37] utilized in the cloud. For example, MASK could be integrated with SRv6 [38] to verify the source and the segment node efficiently.

The integration means that the mechanism could benefit from the security characteristic and the efficiency provided by the Segment Routing.

B. Limitations

We used the link information as one of the MAC inputs to calculate the mark to prevent two colluding attackers from sharing the keys. But consider that two colluding adjacent routers (R_{i-1}, R_{i+1}) on the anticipated path forward the packet to another colluding router (R_i) not on the anticipated path. Even every packet verified at each hop couldn't discover it since R_{i-1} and R_{i+1} could calculate the marks pass any entities' verification [7]. In this case, the mechanism such as MINOS [39] could supervise the action of the routers, which ensures the routers only perform the anticipated action sequence, could discover the misbehavior. But it needs more complex operation than source and path verification. MASK could utilize MINOS to evaluate routers' reputation and then select routers with a high reputation for data transmission.

VIII. Related Work

Path availability. ACR [3] utilizes the availability providers (APs) to provide the edge with multiple routes for each destination. It efficiently provides communication availability. Other path-aware Internet, such as NEBULA [4], SCION [5] [6], to give transparency and choices to end hosts, while SR [40] provides a more practical way to program a dedicated network. These mechanisms are orthogonal with MASK, provide the basis of path verification by allowing the hosts to embed a path of their choice in the packet header.

AS-Key negotiation. Passport [24] enables any two ASes to compute a shared secret based on Diffie-Hellman key exchange protocol. APPA [14] could also negotiate the key among ASes based on the state machine. PISKES [11] establishes a hierarchical symmetric key system. They're orthogonal with our design. We could utilize KDSes to share the Multi-AS-Key, then derive and distribute the session keys to the relative entities.

Verification in dedicated network. vSFC [37] achieves the verification in the cloud to verify the real enforcement of service function chains as expected, extending the verification to a dedicated network. PSVM [41] utilizes Credible Guarantee Agent (CGA) in an SDN network to perform a flexible source and path verification.

Source and path verification. Some path identification mechanisms like Pi [42] and SNAPP [43] utilize path identifier to detect the DDoS attack or pin the forwarding path, could achieve the path verification, but the identifier could be easily forged. In ICING [44], the router computes a MAC for each downstream router on the path. It requires each router to compute $n + 1$ MAC operations per packet. OPT [7] utilizes the DRKey to share the session keys between the hosts and the routers. RFL [28] could further achieve the verification on unreliable communication channels. EPIC [9] proposes a sophisticated mechanism to improve the security and efficiency of the verification. It provides a hierarchical security level for different security requirements. But these mechanisms incur heavy computation overhead since they operate the MAC operation at each hop, while the timestamp synchronization between the routers and end hosts is also impractical. PPV [10] reduces both computation and communication overhead by probabilistic marking per packet, but it couldn't achieve the source verification in the router. Meanwhile, OSV [45] creates a more efficient system by using orthogonal sequences instead of the cryptographic operation. In MASK, the verification is just like a service toward the end host, which balanced the core router's overhead and the end host's verification requirement.

IX. Conclusions

Existing systems for source and path verification faced a dilemma between security and efficiency in computation and communication overhead. Our efficient MASK resolves this dilemma. In contrast to performing per-packet verification in each entity, MASK offers a dramatically different design point by leveraging the end host's ability to establish a per-session verification, ensuring security and reducing overhead in the router, as well as provides the customized capability to meet the end host's requirements. Thus suits the Internet inter-domain communication, and facilitates the security guarantee in a dedicated network that could customize its relative policy. In a word, MASK enables all on-path routers and the destination to achieve the source and path verification in the data plane efficiently, thus provides a secure foundation for the data plane in the network layer.

Acknowledgment

The authors would like to thank anonymous reviewers for their valuable comments. This work was in part supported by the National Key R&D Program of China with No. 2018YFB1800402, No. 2018YFB0803405, China National Funds for Distinguished Young Scientists with No. 61825204, National Natural Science Foundation of China with No. 61932016, Beijing Outstanding Young Scientist Program with No. BJJWZYJH01201910003011, BNRist with No. BNR2019RC01011, PCL Future Greater-Bay Area Network Facilities for Largescale Experiments and Applications with No. LZC0019. Ke Xu is the corresponding author of this paper.

References

[1] X. Feng, C. Fu, Q. Li, K. Sun, and K. Xu, "Off-path TCP exploits of the mixed IPID assignment," in *CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, November 9-13, 2020*, pp. 1323–1335, 2020.

[2] H. Birge-Lee, Y. Sun, A. Edmundson, J. Rexford, and P. Mittal, "Bamboozling certificate authorities with BGP," in *27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018*, pp. 833–849, 2018.

[3] D. Wendlandt, I. C. Avramopoulos, D. G. Andersen, and J. Rexford, "Don't secure routing protocols, secure data delivery," in *5th ACM Workshop on Hot Topics in Networks - HotNets-V, Irvine, California, USA, November 29-30*, 2006.

[4] T. Anderson, K. Birman, R. M. Broberg, M. Caesar, D. Comer, C. Cotton, M. J. Freedman, A. Haeberlen, Z. G. Ives, A. Krishnamurthy, W. H. Lehr, B. T. Loo, D. Mazières, A. Nicolosi, J. M. Smith, I. Stoica, R. van Renesse, M. Walfish, H. Weatherspoon, and C. S. Yoo, "The NEBULA future internet architecture," in *The Future Internet - Future Internet Assembly 2013: Validated Results and New Horizons*, pp. 16–26, 2013.

[5] X. Zhang, H. Hsiao, G. Hasker, H. Chan, A. Perrig, and D. G. Andersen, "SCION: scalability, control, and isolation on next-generation networks," in *32nd IEEE Symposium on Security and Privacy, S&P 2011, 22-25 May 2011, Berkeley, California, USA*, pp. 212–227, 2011.

[6] J. Kwon, J. A. García-Pardo, M. Legner, F. Wirz, M. Frei, D. Hausheer, and A. Perrig, "SCIONLAB: A next-generation internet testbed," in *28th IEEE International Conference on Network Protocols, ICNP 2020, Madrid, Spain, October 13-16, 2020*, pp. 1–12, 2020.

[7] T. H. Kim, C. Basescu, L. Jia, S. B. Lee, Y. Hu, and A. Perrig, "Lightweight source authentication and path validation," in *ACM SIGCOMM 2014 Conference, SIGCOMM'14, Chicago, IL, USA, August 17-22, 2014*, pp. 271–282, 2014.

[8] X. Liu, A. Li, X. Yang, and D. Wetherall, "Passport: Secure and adoptable source authentication," in *5th USENIX Symposium on Networked Systems Design & Implementation, NSDI 2008, April 16-18, 2008, San Francisco, CA, USA, Proceedings*, pp. 365–376, 2008.

[9] M. Legner, T. Klenze, M. Wyss, C. Sprenger, and A. Perrig, "EPIC: every packet is checked in the data plane of a path-aware internet," in *29th USENIX Security Symposium, USENIX Security 2020, August 12-14, 2020*, pp. 541–558, 2020.

[10] B. Wu, K. Xu, Q. Li, Z. Liu, Y. Hu, M. J. Reed, M. Shen, and F. Yang, "Enabling efficient source and path verification via probabilistic packet marking," in *26th IEEE/ACM International Symposium on Quality of Service, IWQoS 2018, Banff, AB, Canada, June 4-6, 2018*, pp. 1–10, 2018.

[11] B. Rothenberger, D. Roos, M. Legner, and A. Perrig, "PISKES: pragmatic internet-scale key-establishment system," in *ASIA CCS '20: The 15th ACM Asia Conference on Computer and Communications Security, Taipei, Taiwan, October 5-9, 2020*, pp. 73–86, 2020.

[12] D. L. Mills, J. Martin, J. L. Burbank, and W. T. Kasch, "Network time protocol version 4: Protocol and algorithms specification," *RFC*, vol. 5905, pp. 1–110, 2010.

[13] B. Haberman and D. L. Mills, "Network time protocol version 4: Autokey specification," *RFC*, vol. 5906, pp. 1–58, 2010.

[14] B. I. Jun, L. Bingyang, W. U. Jianping, and S. Yan, "Preventing ip source address spoofing: A two-level, state machine-based method," *Tsinghua Science and Technology*, vol. 014, no. 004, pp. 413–422, 2009.

[15] D. Dolev and A. C. Yao, "On the security of public key protocols," *IEEE Trans. Inf. Theory*, vol. 29, no. 2, pp. 198–207, 1983.

[16] B. Barak, S. Goldberg, and D. Xiao, "Protocols and lower bounds for failure localization in the internet," in *Advances in Cryptology - EUROCRYPT 2008, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings*, pp. 341–360, 2008.

[17] A. Yaar, A. Perrig, and D. X. Song, "SIFF: A stateless internet flow filter to mitigate ddos flooding attacks," in *2004 IEEE Symposium on Security and Privacy (S&P 2004), 9-12 May 2004, Berkeley, CA, USA*, p. 130, 2004.

[18] "Resource reports." [EB/OL]. https://labs.apnic.net/dists/ascc.html, 2021.

[19] J. Wu, J. Bi, M. Bagnulo, F. Baker, and C. Vogt, "Source address validation improvement (SAVI) framework," *RFC*, vol. 7039, pp. 1–14, 2013.

[20] J. Wu, G. Ren, and X. Li, "Source address validation: Architecture and protocol design," in *Proceedings of the IEEE International Conference on Network Protocols, ICNP 2007, October 16-19, 2007, Beijing, China*, pp. 276–283, 2007.

[21] G. Hu, K. Xu, J. Wu, Y. Cui, and F. Shi, "A general framework of source address validation and traceback for ipv4/ipv6 transition scenarios," *IEEE Netw.*, vol. 27, no. 6, pp. 66–73, 2013.

[22] Y. Hu, M. Jakobsson, and A. Perrig, "Efficient constructions for one-way hash chains," in *Applied Cryptography and Network Security, Third International Conference, ACNS 2005, New York, NY, USA, June 7-10, 2005, Proceedings*, pp. 423–441, 2005.

[23] M. Zhang, G. Li, S. Wang, C. Liu, A. Chen, H. Hu, G. Gu, Q. Li, M. Xu, and J. Wu, "Poseidon: Mitigating volumetric ddos attacks with programmable switches," in *27th Annual Network and Distributed System Security Symposium, NDSS 2020, San Diego, California, USA, February 23-26, 2020*, 2020.

[24] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese, and D. Walker, "P4: programming protocol-independent packet processors," *Comput. Commun. Rev.*, vol. 44, no. 3, pp. 87–95, 2014.

[25] N. Foster, N. McKeown, J. Rexford, G. Parulkar, L. Peterson, and O. Sunay, "Using deep programmability to put network owners in control," *SIGCOMM Comput. Commun. Rev.*, vol. 50, p. 82–88, Oct. 2020.

[26] M. Hogan, S. L. Feibish, M. T. Arashloo, J. Rexford, D. Walker, and R. Harrison, "Elastic switch programming with p4all," in *HotNets '20: The 19th ACM Workshop on Hot Topics in Networks, Virtual Event, USA, November 4-6, 2020*, pp. 168–174, 2020.

[27] S. E. Deering and R. M. Hinden, "Internet protocol, version 6 (ipv6) specification," *RFC*, vol. 8200, pp. 1–42, 2017.

[28] B. Wu, K. Xu, Q. Li, B. Liu, S. Ren, F. Yang, M. Shen, and K. Ren, "RFL: robust fault localization on unreliable communication channels," *Comput. Networks*, vol. 158, pp. 158–174, 2019.

[29] V. Paxson, "End-to-end routing behavior in the internet," *Comput. Commun. Rev.*, vol. 36, no. 5, pp. 41–56, 2006.

[30] "As6447 bgp routing table analysis report." [EB/OL]. https://bgp.potaroo.net/as6447/, 2021.

[31] B. Liu, J. T. Chiang, J. J. Haas, and Y. Hu, "Coward attacks in vehicular networks," *ACM SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 14, no. 3, pp. 34–36, 2010.

[32] S. Yao, Z. Li, J. Guan, and Y. Liu, "Stochastic cost minimization mechanism based on identifier network for iot security," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 3923–3934, 2020.

[33] S. Yao, J. Guan, Y. Wu, K. Xu, and M. Xu, "Toward secure and lightweight access authentication in sagins," *IEEE Wirel. Commun.*, vol. 27, no. 6, pp. 75–81, 2020.

[34] S. Yao, J. Guan, Z. Yan, and K. Xu, "SI-STIN: A smart identifier framework for space and terrestrial integrated network," *IEEE Netw.*, vol. 33, no. 1, pp. 8–14, 2018.

[35] X. Chen, "Implementing AES encryption on programmable switches via scrambled lookup tables," in *Proceedings of the 2020 ACM SIGCOMM 2020 Workshop on Secure Programmable Network Infrastructure, SPIN@SIGCOMM 2020, Virtual Event, USA, August 14, 2020*, pp. 8–14, 2020.

[36] P. Zave and J. Rexford, "The compositional architecture of the internet," *Commun. ACM*, vol. 62, no. 3, pp. 78–87, 2019.

[37] X. Zhang, Q. Li, J. Wu, and J. Yang, "Generic and agile service function chain verification on cloud," in *25th IEEE/ACM International Symposium on Quality of Service, IWQoS 2017, Vilanova i la Geltrú, Spain, June 14-16, 2017*, pp. 1–10, 2017.

[38] C. Filsfils, S. Previdi, L. Ginsberg, B. Decraene, S. Litkowski, and R. Shakir, "Segment routing architecture," *RFC*, vol. 8402, pp. 1–32, 2018.

[39] L. Xu, K. Xu, M. Shen, K. Ren, J. Fan, C. Guan, and W. Chen, "MINOS: regulating router dataplane actions in dynamic runtime environments," in *Proceedings of the ACM Turing 50th Celebration Conference - China, TUR-C 2017, Shanghai, China, May 12-14, 2017*, pp. 40:1–40:10, 2017.

[40] R. Hartert, S. Vissicchio, P. Schaus, O. Bonaventure, C. Filsfils, T. Telkamp, and P. François, "A declarative and expressive approach to control forwarding paths in carrier-grade networks," in *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication, SIGCOMM 2015, London, United Kingdom, August 17-21, 2015*, pp. 15–28, 2015.

[41] F. Yang, K. Xu, Q. Li, R. Lu, B. Wu, T. Zhang, Y. Zhao, and M. Shen, "I know if the journey changes: Flexible source and path validation," in *28th IEEE/ACM International Symposium on Quality of Service, IWQoS 2020, Hangzhou, China, June 15-17, 2020*, pp. 1–6, 2020.

[42] A. Yaar, A. Perrig, and D. X. Song, "Pi: A path identification mechanism to defend against ddos attack," in *2003 IEEE Symposium on Security and Privacy (S&P 2003), 11-14 May 2003, Berkeley, CA, USA*, p. 93, 2003.

[43] B. Parno, A. Perrig, and D. G. Andersen, "SNAPP: stateless network-authenticated path pinning," in *Proceedings of the 2008 ACM Symposium on Information, Computer and Communications Security, ASIACCS 2008, Tokyo, Japan, March 18-20, 2008*, pp. 168–178, 2008.

[44] J. Naous, M. Walfish, A. Nicolosi, D. Mazières, M. Miller, and A. Seehra, "Verifying and enforcing network paths with icing," in *Proceedings of the 2011 Conference on Emerging Networking Experiments and Technologies, Co-NEXT '11, Tokyo, Japan, December 6-9, 2011*, p. 30, 2011.

[45] H. Cai and T. Wolf, "Source authentication and path validation with orthogonal network capabilities," in *2015 IEEE Conference on Computer Communications Workshops, INFOCOM Workshops, Hong Kong, China, April 26 - May 1, 2015*, pp. 111–112, 2015.