Enabling Efficient Source and Path Verification via Probabilistic Packet Marking

Bo Wu†‡, Ke Xu†‡, Qi Li‡†, Zhuotao Liu§, Yih-Chun Hu‡, Martin J. Reed‡, Meng Shen‖, Fan Yang‡

†Department of Computer Science and Technology, Tsinghua University, Beijing, China
‡Graduate School at Shenzhen, Tsinghua University, Shenzhen, China
§School of Computer Science, Beijing Institute of Technology, Beijing, China
‖School of Computer Science, University of Essex, Colchester, UK
‡School of Computer Science and Electronic Engineering, University of Essex, Colchester, UK
†Graduate School at Shenzhen, Tsinghua University, Shenzhen, China

corresponding address:

Emails: {wub14@mails.tsinghua.edu.cn, xuke@tsinghua.edu.cn, qi.li@sz.tsinghua.edu.cn, zliu48@illinois.edu, yihchun@illinois.edu, mjreed@essex.ac.uk, shemeng@bit.edu.cn, y-f14@tsinghua.org.cn}

Abstract—The Internet lacks verification of source authenticity and path compliance between the planned packet delivery paths and the real delivery paths, which allows attackers to construct attacks like source spoofing and traffic hijacking attacks. Thus, it is essential to enable source and path verification in networks to detect forwarding anomalies and ensure correct packet delivery. However, most of the existing security mechanisms can only capture anomalies but are unable to locate the detected anomalies. Besides, they incur significant computation and communication overhead, which exacerbates the packet delivery performance. In this paper, we propose a high-efficient packet forwarding verification mechanism called PPV for networks, which verifies packet source and their forwarding paths in real time. PPV enables probabilistic packet marking in routers instead of verifying all packets. Thus, it can efficiently identify forwarding anomalies by verifying markings. Moreover, it localizes packet forwarding anomalies, e.g., malicious routers, by reconstructing packet forwarding paths based on the packet markings. We implement PPV prototype in Click routers and commodity servers, and conducts real experiments in a real testbed built upon the prototype. The experimental results demonstrate the efficiency and performance of PPV. In particular, PPV significantly improves the throughput and the goodput of forwarding verification, and achieves around 2 times and 3 times improvement compared with the state-of-art OPT scheme, respectively.

Index Terms—Source and Path Verification, Fault Localization

I. INTRODUCTION

The current Internet is vulnerable to various types of malicious attacks, such as source spoofing and traffic hijacking attacks, because it lacks source authentication and path validation [1] [2] [3]. All network entities (e.g., packet source, intermediate router, and destination) assume that the incoming packets are authentic although they may be counterfeited or modified [4] [5] [6]. Additionally, the connectionless nature of IP packet delivery cannot ensure path compliance between actual forwarding paths in the data-plane and the computed paths in the control-plane [7]. Therefore, source authentication and path validation are essential in IP network to detect packet forwarding anomalies and ensure correct packet delivery. Here, source authentication verifies whether a received packet indeed originates from the real sender claimed by the source address of IP packet, and path validation confirms whether the actual forwarding path taken by traffic [8] is consistent with the paths selected by the source, or computed according to the routing policies of an ISP, enterprise, or datacenter network.

Efficiency is critical for the practical deployment of Internet security protocols [9]. Unfortunately, existing schemes for source and path verification impose a non-trivial overhead since they require all intermediate entities to perform cryptographic operations upon receiving packets. These operations (e.g., sign and verification) bring a significant verification overhead. For example, prior AS-level packet verification mechanisms [1] [10] [11] verify every forwarded packet hop-by-hop during its transmission. Such hop-by-hop processing could significantly reduce the forwarding efficiency (e.g., throughput and goodput) of the network. As packets can also be hijacked due to attacks on systems such as the OSPF protocol [12] [13], several router-level source and path verification approaches [2] [3] [14] [15] have been proposed. However, they all require the source to initialize markings in packet headers for all intermediate entities, which imposes higher communication overhead. Meanwhile, each intermediate entity is required to verify these markings once receiving a packet, adding additional verification overhead and potentially reducing the quality of service in networks. Besides the high overhead, prior approaches can only capture forwarding anomalies but are unable to localize these anomalies. In particular, the intermediate entities will directly drop the packets once any anomaly is detected, while no feedback is sent to end-hosts. Without such verification feedback from entities, the end-hosts cannot localize a fault during packet transmission, which is not conducive to network repair and management.

In this paper, we propose PPV, a highly efficient mechanism for verifying source authenticity and path compliance. In PPV, entities perform probabilistic packet marking for each packet and generate a key-hashed Message Authentication Code (MAC) for that packet. We show that by using PPV each packet only requires to be marked by at most two entities (rather than all entities), thereby greatly reducing the computation overhead of intermediate entities and the verification overhead of the destination. Through the evaluation on our prototype, PPV introduces a modest 6.25% mean commu-
In a network, a packet source can be spoofed and forwarding policies, which makes actual forwarding path (denoted by \Psi) differ from intended forwarding path (denoted by \Psi). With \Psi = \langle R_1, R_2, R_3 \rangle in Fig. 1, we summarize various types of path inconsistency as following descriptions.

**Routing addition.** The packets are abnormally forwarded to other entities out of \Psi and then delivered to \Psi, such as the additional \( R_6 \) between \( R_2 \) and \( R_3 \) in Fig. 1.

**Routing skipping.** The packets skip one or more entities and are instead forwarded to other downstream entities on \Psi, such as the delivery from \( R_1 \) to \( R_3 \) (ignoring \( R_2 \)) in Fig. 1.

**Unordered routing path.** The packets are delivered through the entities that are same but out-of-order compared with entities on \Psi, such as \( \Phi = \langle R_1, R_3, R_2, R_4 \rangle \) in Fig. 1.

**Routing detour.** The packets are maliciously forwarded away from \Psi and then deliberately back to \Psi after a short travel, e.g., the delivery from \( R_2 \) to \( R_6 \), \( R_7 \), and then to \( R_4 \) in Fig. 1.

**Path complete deviation.** The packets are delivered through entities that are totally different from the entities on \Psi, such as \( \Phi = \langle R_5, R_6, R_7, R_8 \rangle \) in Fig. 1.

**B. Sophisticated Attacks**

In addition to the above, there are more advanced attacks that should be taken into consideration when guaranteeing network security with regard to verifying the source and path.

**Combined attack.** An adversary can control multiple compromised entities for launching combined attacks, which may consist of several types of the basic attacks described above.

**Routing collusion.** The compromised entities can collude with each other to destroy the veracity of the source and the path so that they can launch any of the basic attacks together.

**Fault location hidden.** An adversary can make the unreliable sender or malicious router(s) try to hide their location to obviate the source and path verification.

### III. PPV DESIGN OVERVIEW

In this section, we will present a high-level description of the proposed PPV that performs source and path verification.

#### A. Scope and Assumptions

This paper aims to verify packet source and forwarding path of end-to-end communication in the data plane. We assume that the existing secure routing protocols can enable end-hosts to learn the intended forwarding path \( \Psi \) at AS- or router-level granularity. For example, the AS-level path can be obtained through some BGP related protocols [1] [19]; whereas, SCION [20] or Pathlet routing [21] can make the source specify the router-level forwarding path in the packet header. The end-hosts can also obtain the necessary symmetric keys, shared with intermediate entities of the intended path, using the existing DRKey protocols [2]. DRKey uses stateless operation in entities and does not require each entity to store the symmetric keys, avoiding DoS attack by state exhaustion.

Our proposed scheme, PPV, works at a flow-level granularity so as to achieve source authentication and path validation. It is unnecessary to perform strong per-packet forwarding verification.
verification to achieve perfect detection as source spoofing and path inconsistency of one single packet cannot, usually, pose a serious threat to the connection [22]. Instead, flow-level forwarding verification can greatly reduce the overhead compared with the per-packet verification schemes.

B. Design Challenges

We highlight the design challenges of source and path verification by introducing an intuitive but insecure strawman approach that is built upon an IP traceback scheme. Let us consider the source $S$ communicates with the destination $D$. Before each packet departs, $S$ reserves several fields in packet header. During packet delivery, entities will sign its address on theses fields in order. $D$ will perform the packet verification according to this address sequence. However, an offending entity can disturb the verification of $D$, by such signing a counterfeit address or tampering some signed addresses. Without encryption techniques, the entities are all open to counterfeiting or modification attacks [23]. However, using encryption techniques can introduce non-negligible overhead, such as higher verification overhead for encrypted markings, which impacts the forwarding efficiency. Thus, there is a tradeoff between higher security and lower overhead. i) Simply using the entity address as a marking occupies few bytes and brings a lower overhead, but it is open to marking modification and counterfeiting attacks. ii) Using cryptographic techniques can strengthen security but with increased overhead, lowering the forwarding efficiency, such as throughput and goodput.

C. PPV Overview

PPV provides probabilistic packet marking for each entity, bridging the gap between security and performance. PPV enables each packet to record the markings of a subset of the entities instead of all entities on forwarding path, introducing a lower communication overhead. This also reduces the overall packet verification overhead as every packet is not marked by all entities. PPV eschews the use of a simple address as a marking, but instead improves security by employing a cryptographic marking based on a MAC using symmetric keys.

In the network, the forwarding path $P$, including $\Psi$ and $\Phi$, can be considered as the composition of many links $L_i$ between two entities as Eq. 1 shows. Each link $L_i$ is determined by two neighbor entities, such as $L_0$ for $S$ and $R_1$, $L_i$ for $R_i$ and $R_{i+1}$, and $L_n$ for $R_n$ and $D$.

$$Path \equiv \langle L_0, L_1, \ldots , L_i, \ldots , L_n \rangle.$$ (1)

PPV aims to make each packet collect one connection $L_i$ instead of all connections during its travel from $S$ to $D$. Thus, only two fields should be reserved in a packet by $S$, which minimizes communication overhead. The two reserved fields are designed to store two markings of neighbored entities. The high-level PPV operations are sketched in Fig. 2.

**PPV overview**. Different entities handle the packets according to the responding rules.

PPV aims to make each packet collect one connection $L_i$ instead of all connections during its travel from $S$ to $D$. Thus, only two fields should be reserved in a packet by $S$, which minimizes communication overhead. The two reserved fields are designed to store two markings of neighbored entities. The high-level PPV operations are sketched in Fig. 2.

**PPV header initialization.** Before packet’s departure, $S$ first creates a new PPV header between Layer 3 (IP header) and Layer 4 (TCP header) as Fig. 3 shows. The seven fields in PPV header are then all initialized for later marking and verification.

**Probabilistic packet marking.** On receiving a packet, two neighbored entities probabilistically make encrypted markings on PPV header. In PPV, the marking probability of each entity is well designed for reduced convergence delay when the destination performs packet verification.

**Packet pre-authentication.** The packet pre-authentication is mainly used to verify the integrity of packet data and the validity of markings, which protects the packet data or markings in PPV header from modification or counterfeit.

**Source and path verification.** In PPV, each packet records two markings of neighbor entities, and the destination can check these markings. If fails, it illustrates the source or the forwarding path has been modified by some malicious entity.

**Path reconstruction and fault localization.** If the verification fails, $D$ will reconstruct $\Psi$ to localize the fault, using the collected connections $L_i$, which are all obtained from a series of the received PPV headers. The comparison between $\Phi$ and $\Psi$ will contribute to localizing the offending entity.

IV. THE DESIGN DETAILS OF PPV

In this section, we detail our proposed PPV to implement end-to-end forwarding verification. For the packet delivery, the sender first initializes PPV header (Section IV-A), intermediate entities perform probabilistic packet marking (Section IV-B), and the destination carries out packet pre-authentication (Section IV-C), source and path verification (Section IV-D), and fault localization (Section IV-E).

A. **PPV header initialization**

PPV header is used to store markings made by intermediate entities during packet transmission, which is employed for packet forwarding verification. For an outgoing packet, the PPV header is created between Layer 3 and Layer 4, as shown in Fig. 3. It contains seven fields: FlowID, PacketID, Marking Verification Field (MVF), two address fields (Add$_1$ and Add$_2$) and two marking fields (MF$_1$ and MF$_2$).
FlowID is an identifier of network traffic, calculated with hash over six elements: addresses of end-hosts (src and dst), ports of end-hosts (srcp and dstp), protocol (prtI), and the hash value of $K_{SD}$ shared between $S$ and $D$, as shown in Eq. 2. FlowID can also mitigate source spoofing as the packet sender does not have the valid $K_{SD}$.

\[
FlowID = H([src||srcp||dst||dstp||prtI||H(K_{SD})]). \tag{2}
\]

PacketID is a packet identifier, which is the MAC value of FlowID and packet payload keyed with $K_{SD}$, see Eq. 3. PacketID can be used to perform data authentication and prevent replay attacks.

\[
PacketID = MAC_{K_{SD}}(FlowID||payload). \tag{3}
\]

The field MVF is used to prevent the markings in $MF_1$ and $MF_2$ from being modified and counterfeited during packet transmission. $S$ initializes MVF as the value $mvf_0$ (see Eq. 4). Note that $mvf_0$ varies with different packets and can be calculated only by $S$ and $D$ using $K_{SD}$.

\[
mvf_0 = MAC_{K_{SD}}(PacketID). \tag{4}
\]

The fields $Add_1$ and $Add_2$ record the addresses of two neighboring entities that insert their own markings in $MF_1$ and $MF_2$, respectively. Two marking fields have two states: empty (filled with all zeros), or full (otherwise). Initially, they are all empty, representing no entity marks in PPV header.

**B. Probabilistic Packet Marking of Routers**

As stated earlier, the intermediate entities perform probabilistic packet marking to reduce encryption overhead. We respectively define $P$, $C$ and $N$ as the address of previous, current and next entity during packet transmission. On receiving a packet, $R_i$ will first check the status of $MF_1$ and $MF_2$. If they are all empty, $R_i$ will mark its marking $M_i$ on $MF_1$ according to its marking probability $P_i$ (described shortly). The marking $M_i$ is the MAC operation value of src and TTL (time-to-live), $P$ and PacketID keyed with $K_i$, as Eq. 5 shows. The source address src is an essential input of MAC operation, by including it, if a malicious entity launches source spoofing, this will cause an error in the marking of downstream entity and a failure of source authentication allowing detection.

\[
M_i = MAC_K(src||TTL||P||PacketID). \tag{5}
\]

Meanwhile, $Add_1$ and $Add_2$ are respectively written as $C$ (i.e., $R_j$) and $N$ (i.e., $R_{i+1}$). $R_i$ performs the MAC operation to update MVF from $mvf_0$ to $mvf_1$ as follows.

\[
mvf_1 = MAC_K(mvf_0||C||M_i||N). \tag{6}
\]

On receiving the packet, $R_{i+1}$ finds $MF_1$ is full and the addresses on the fields $Add1$ and $Add2$ are respectively equal to the addresses of $R_i$ and $R_{i+1}$. Then, $R_{i+1}$ inserts its marking $M_{i+1}$ into $MF_2$ and updates MVF to $mvf_2$ as shown in Eq. 7.

\[
mvf_2 = MAC_{K_{i+1}}(mvf_1||M_{i+1}). \tag{7}
\]

By using DRKey protocol [2], each entity dynamically recreates the symmetric key on the fly and does not store it for each flow or source, which protects entities from different types of state exhaustion DoS attacks. In PPV, the intermediate entity will drop the packet if it finds the PPV header contains the full $Add_2$, but not filled with its own address when $MF_2$ is empty. Fig. 4 shows one example of the process of probabilistic packet marking when a packet is forwarded from $S$ to $D$. In this example, only $R_2$ and $R_3$ make markings in PPV header according to their marking probabilities.

**Marking probability.** To verify the packet forwarding on $Ψ$, PPV enables $D$ to quickly obtain all markings by setting the marking probability $P_i$ for each entity $R_i$. We define $Q_i$ as the probability that $R_i$’s marking appears on $MF_1$, which is calculated by Eq. 8.

\[
Q_i = (1 - P_1)(1 - P_2)...(1 - P_{i-1})P_i. \tag{8}
\]

UPPM [24] has proved equal appearing probability, i.e., $Q_1 = Q_2 = ... = Q_n$ can bring the minimum convergence time. In PPV, we set $P_i$ using the TTL value as: $P_i = TTL^i$. PPV can set the value of $P_i$ by adjusting the TTL value a in packet header when the packets depart from $S$. Next, we will provide the proof for $Q_1 = Q_2 = ... = Q_3$ under the setting of $P_i$. We denote $\chi (\chi \geq n)$ as the initial TTL value when the packets depart from $S$. Based on Eq. 8, $Q_3$ can be calculated as follows, demonstrating $Q_i$ is a constant value.

\[
Q_i = \frac{\chi - 1}{\chi}, \frac{\chi - 2}{\chi}, ..., \frac{1}{\chi} = \frac{1}{\chi}. \tag{9}
\]

Thus, all marks have the same probability of appearing at $D$ using $P_i$, i.e., $Q_1 = Q_2 = ... = Q_3$.

We denote $P_0$ as the value of Eq. 9. Thus, $P_0$ can be presented by Eq. 10. On receiving a packet, each entity can calculate its own marking probability $P_i$ regardless of the number of flows and path lengths. Note that computing $P_i$ causes a smaller marking probability of entities close to $S$, to bring an equal $Q_i$ of all marks, overcoming the otherwise bias of hops at the beginning of the path which equal probability marking schemes suffer from.

\[
P_0 = \frac{1}{\chi}, \quad 0 \leq P_0 \leq \frac{1}{n}. \tag{10}
\]

The probability that $R_{i+1}$ marks on $MF_2$ only depends on whether a previous entity $R_i$ has inserted $R_{i+1}$’s address into $Add_2$, rather than the probability $P_{i+1}$. As for $P_0$, we define standard probability (SP) when $P_0 = 1/n$, in which two markings of at least one pair of entities will be certainly recorded in PPV header, because $P_i$ tends to $1$ when packets are close to $D$. More especially, $P_0 = 1$ illustrates $R_0$ certainly marks on $MF_1$ if none of upstream entities have marked in PPV header. Note that the marking probability $P_i$ depends on the current TTL value. In this case, an offending entity, say $R_α$, can launch a TTL attack to confuse TTL value, and make the marking probabilities of downstream entities abnormal. For example, $R_α$ can either make TTL value larger to bring a smaller $P_{α+1}$, or decrease TTL value greatly to make $R_{α+1}$
mark certainly, PPV can avoid this TTL attack by using the value of TTL to calculate $R_1$’s marking $M_1$ as Eq. 5 shows. If the TTL value on IP header is maliciously modified, the downstream entities will generate erroneous markings, causing the failure of later verification.

**C. Packet Pre-authentication**

The packet pre-authentication includes data authentication, which verifies the integrity of packet data, and the marking validation, ensuring that the received PPV header is reliable.

**Packet data authentication.** On receiving each packet, $D$ first performs data authentication, because if the packet data is modified, it is meaningless to verify the source address and forwarding path. The field PacketID is employed to achieve this requirement. Specifically, $D$ first recomputes the value of PacketID using Eq. 3 and then compares it with the received PacketID in PPV header. With passing the data authentication, the following marking verification can be carried out.

**Marking validity authentication.** The markings on $MF_1$ and $MF_2$ are used by PPV for source and path verification. With symmetric keys, $D$ recomputes $mfv_2$, the value of MVF, to verify whether the markings on the two marking fields have been modified. The equivalence between the recalculated $mfv_2$ and the received $mfv_2$ in the PPV header illustrates the markings on two marking fields are valid and have not been modified. $D$ will employ the valid markings for later verification and fault localization.

**D. Source Authentication and Path Validation**

Source and path verification aims to capture anomalies during packet transmission. After packet pre-authentication, $D$ then carries out source and path verification. Due to the probabilistic packet marking process, $MF_1$ and $MF_2$ may be empty when the packet arrives at $D$, with the probability $P_e$:

$$P_e = \prod_{i=1}^{n} (1 - P_i) = 1 - n \cdot P_0, \quad 0 \leq P_0 \leq \frac{1}{n}. \quad (11)$$

If this is the case, $D$ will ignore this verification of source and path, and verify the next marked packet. From Eq. 11, we know if $P_0$ takes the value of $SP$, i.e., $P_0 = 1/n$, $D$ will never receive an empty marking field in PPV header.

Algorithm 1 shows the process of source authentication and path validation. According to the addresses $R_1$ and $R_2$ on the fields $Add_1$ and $Add_2$, $D$ will use the respective symmetric keys $K_1$ and $K_2$ to recalculate the markings: $M'_1$ and $M'_2$. Then the comparison between recalculated $M'_1$ ($M'_2$) and received $M_1$ ($M_2$) in PPV header will determine whether source and path verification succeeds. If unequal, it illustrates that the source address and/or the forwarding path have been modified.

**Confidence degree.** For source and path validation, it is unnecessary to perform strong per-packet verification, as a single packet is unlikely to pose a threat. In this paper, PPV works on flow granularity, in which the marked packet can be used to verify the path integrity of two adjacent entities. In this case, one successful verified packet cannot represent the reliability of the entire forwarding path. We employ a parameter called confidence degree $C$ to show the credibility of the whole forwarding path where $C$ is the proportion between the reliable entities and all forwarding entities on the path after validating a certain number of packets. If the forwarding path contains $n$ entities, there are $n$ connections ignoring the first connection (i.e., $L_0$) between $S$ and $R_1$. The connection $L_i$ is uniquely determined by $R_i$ and $R_{i+1}$, which can be found on $Add_1$ and $Add_2$. Thus, the confidence degree $C$ can be evaluated by the proportion of legal verified connections. We use $X_i$ as a decision variable, denoting whether connection $L_i$ is verified, that is to say whether $M_i$ and $M_{i+1}$ have all passed the validation after $m$ packets’ arriving. Thus, $X_i$ is:

$$X_i = \begin{cases} 1 & L_i \text{ is verified after receiving } m \text{ packets} \\ 0 & \text{otherwise} \end{cases} \quad (12)$$

We define $X$ as the sum of $X_i$, i.e., $X = \sum_{i=1}^{n} X_i$. Using the known marking probability, we know that the expectation of the number of marked packets is $mnP_0$ after receiving $m$ packets. Among these marked packets, every connection $L_i$ has the equal probability $1/n$ to appear on PPV header (see Section IV-B). So the probability that $L_i$ does (not) appear on PPV header, i.e., is (not) validated by $D$ is:

$$P[X_i] = \begin{cases} \frac{1 - (1 - 1/n)^{mnP_0}}{1 - (1 - 1/n)^{mnP_0}} & X_i = 0 \\ 1 & X_i = 1 \end{cases} \quad (13)$$

So the expected number of all verified links is as follows.

$$E(X) = E\left(\sum_{i=1}^{n} X_i\right) = n[1 - (1/n)^{mnP_0}] \quad (14)$$

Then, the confidence degree $C$ is:

$$C = \frac{E(X)}{n} = 1 - \left(1 - \frac{1}{n}\right)^{mnP_0} \quad (15)$$

Fig. 5 illustrates the relationship between the confidence degree $C$ and the number of verified packets $m$ at $D$ when the path length is 5 hops and 13 hops, the average path length in AS [25] and router level [26], respectively. $P_0$ takes the basic probabilities of $SP_{n=5} = 1/5$ and $SP_{n=13} = 1/13$, and a smaller value of 0.05 that is derived from the basic probability of an assumed longest path length i.e., $n = 20$ hops [26]. From this figure, we learn the confidence degree $C$ increases with the number of packets that are verified. For example, for $SP_{n=5}$, the confidence degree of 90%, 95%, 99% and 99.9% only require around 10, 13, 21, 31 verified packets. For the same path length, the larger $P_0$ (e.g., $P_0 = 0.20$) value requires fewer verified packets to reach the same $C$ than the smaller value (e.g., $P_0 = 0.05$). For the longer forwarding path, a greater number of received packets are needed to be verified to gain the same $C$. For example, $C = 99\%$ needs 21 verified packets.
of the flow. Due to a unit, it is proportional to time for a given data rate. We define reconstruct the forwarding path. While Convergence delay.

In our proposed PPV, the scope of the fault can be narrowed down to one or two entities. PPV tries to localize the first offending entity on \( \Psi \), because its misbehavior can lead to the confusion at downstream entities. In other words, it is meaningless to locate all its downstream entities as their apparent misbehavior actually results from the first fault.

**Convergence delay.** In this paper, we define convergence delay \( T_c \) as the expected number of received packets used to reconstruct the forwarding path. While \( T_c \) is not strictly a time unit, it is proportional to time for a given data rate. We define a **useful link** as a link not appearing in the previous packets of the flow. Due to \( Q_c = P_0 \), as described in Section IV-D, the probability of getting a useful connection 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 more generally. Using a similar analysis from other research [16] [24], the derivation of \( T_c \) is a form of the coupon collector problem and is equal to:

\[
T_c = \frac{1}{n \cdot P_0} + \ldots + \frac{1}{(n-j+1) \cdot P_0} + \ldots + \frac{1}{P_0} \\
= \frac{1}{P_0} \sum_{j=1}^{n} \frac{1}{k} = \ln n + W \approx n \cdot P_0, \quad W \approx 0.577.
\]

**Algorithm 2 Fault localization pseudo code.**

1: function FAULT LOCALIZATION ( )
2: Require:
3: \( \Phi = (R_1, R_2, \ldots, R_n), M_1, M_2, \ldots, M_n \)
4: \( \Psi = (R_1, R_2, \ldots, R_n'), M_1', M_2', \ldots, M_n' \)
5: src, srcp, dst, dstp, protcl, KSD, FlowID
6: Compute:
7: \( \text{FlowID}' = H(\text{src}||\text{srcp}||\text{dst}||\text{dstp}||\text{protcl}||KSD) \)
8: if (FlowID == FlowID') then
9: Source host is credible.
10: else
11: Source host is localized for its address spoofing.
12: end if
13: \( l = \max(n, m) \)
14: for \( 0 < t \leq 1 \) do
15: if (\( R_t = R_t' \)) then
16: if (\( M_t = M_t' \)) then
17: Verification succeeds up to \( R_t \)
18: \( \text{else} \)
19: \( R_{t-1} \) and \( R_t \) can be localized.
20: \( \text{end if} \)
21: \( \text{else} \)
22: \( R_{t-1} \) can have changed forwarding path.
23: end if
24: end for
25: end function

**V. SECURITY ANALYSIS**

We argue that PPV provides the verification for both the origin and the forwarding path, and discuss why PPV is resilient to the adversary models described in Section II.

**A. Defense Against Source Address Spoofing**

Without the symmetric key \( KSD \), a malicious sender cannot calculate the valid PacketID, causing an error during packet pre-authentication. Although a malicious sender can use PacketID in other benign packets that have been sent by a reliable sender, \( D \) can also prevent this due to detecting a replay attack through a suitable replay detection cache. Additionally, with a spoofed source, packets carrying two markings will not pass the verification as the spoofed origin will be employed and result in a series of erroneous markings. When an entity launches a spoofing attack by counterfeiting or tampering source address, the packets will be dropped at \( D \) due to the incorrect results of packet pre-authentication.

**B. Defense Against Routing Path Inconsistency**

PPV is sensitive to path inconsistency due to using both the TTL value and the address of a 1-hop upstream entity to compute a valid marking. If there is an entity changing the forwarding path by the means described in Section II-A, the markings of downstream entities will be incorrect, due to the false TTL value and the illegal previous hop address.

For routing addition and routing detour, the packets are forwarded to another entity not included in \( \Psi \) by an offending entity (say \( R_a \)), and then delivered to the entity (say \( R_{a'}, \varphi > \tau \)) on \( \Psi \). Consequently, \( R_a \) will generate an erroneous marking \( M_{a'} \) calculated with an incorrect TTL value and address of the previous entity in \( \Psi \). Additionally, the additional entities no in \( \Psi \) between \( R_a \) and \( R_{a'} \) will also create and insert erroneous markings into the PPV header, which causes the packet to be dropped at \( D \). As for routing skipping, for example, the offending entity \( R_a \) directly forwards the packets to \( R_{a',+e} \) (\( e > 1 \)) on \( \Psi \) and skips the entities between \( R_a \) and \( R_{a',+e} \), in which the marking \( M_{a',+e} \) will be incorrect because of using
incorrect TTL and unintended previous address. In this case, \( R_\tau \) can also maliciously insert \( R_{\tau+1}' \)'s address into \( Add_2 \) for more downstream entities to mark their markings in the PPV header for evading the verification at \( D \). However, PPV enables \( R_{\tau+1} \) to drop the packets once finding a full \( Add_2 \) and an empty \( MF_2 \). For further malicious interference, \( R_\tau \) could also counterfeit markings \( M_{\tau+1} \) on \( MF_2 \) to prevent downstream entities from discarding packets, or steer an already-marked packet on an arbitrary path to \( D \), in which \( D \) can detect the invalid markings via packet pre-authentication and then drop the packets. Similar to the analysis of route skipping, the out-of-order forwarding path can also be addressed in PPV under the basic attack or further interference. If a complete path deviation occurs, \( D \) will drop the packets due to lacking symmetric keys shared with intermediate entities.

C. Defense Against Sophisticated Attacks

For the higher security requirements, there are many sophisticated attacks we should take into consideration when verifying the origin and the forwarding path. PPV can provide strong defense against a combined attack consisting of multiple types of basic attacks. Specifically, if both source spoofing and path inconsistency occur simultaneously, the downstream entities will mark erroneously in the PPV header due to using a modified source address, inaccurate TTL and incorrect previous address as the inputs to the marking calculation. As PPV works at the flow granularity, the offending entity can launch further sophisticated attacks, only aiming at the subset of packets of this flow, which could cause multiple actual forwarding paths. In this case, \( D \) can reconstruct one or more \( \Phi \) and localize the offending entity using Algorithm 2.

As for routing collusion, if there are two malicious entities, \( e.g., R_\tau \) and \( R_\phi \), they can collude with each other to launch sophisticated attacks. Before packets reach \( R_\phi \), \( R_\tau \) can either spoof the source or change the packet forwarding path. On receiving the packet, \( R_\phi \) clears the \( Add_1 \) (\( Add_2 \)) and \( MF_1 \) (\( MF_2 \)) to stop \( D \) from receiving the markings of entities between \( R_\tau \) and \( R_\phi \) for the purpose of hiding \( R_\phi \)'s misbehavior. In this case, the reconstructed \( \Phi \) differs from \( \Psi \), which helps to identify and localize the fault \( R_\tau \). As for attacking fault location, where a hidden adversary tries to control an unreliable sender or entity so as to hide its location for modifying the source or the forwarding path: in PPV, the downstream entities will create erroneous markings, causing a verification failure. With the reconstructed \( \Phi \) and collected markings, \( D \) can localize or narrow down the scope of the fault. To interfere with detection, the adversary could stop the downstream entity from marking the false markings in the PPV header by means of inserting a counterfeit marking. However, without access to the symmetric key, the marking or MVF value will not be correctly calculated or updated, which brings about the failure of packet pre-authentication at \( D \). The adversary could make the packets skip over one or more entities to stop them from marking in the PPV header. However, PPV enables \( D \) to reconstruct the actual forwarding path and then localize this fault.

PPV can also provide defense against modification and counterfeit of the marking. When the adversary modifies or counterfeits the markings of another entity, the packet pre-authentication will fail, because the recomputed MVF is not equal to the received MVF. The marked packets with invalid markings will be dropped by \( D \). A replay attack can also be addressed in PPV. When receiving a packet with the same PacketID with the previously received packet, \( D \) will consider it as a replay attack because PacketID cannot be counterfeited without the corresponding symmetric key.

VI. PERFORMANCE EVALUATION

In this section, we evaluate PPV by using the following important metrics. (i) Packet communication overhead that indicates the cost of forwarding verification. (ii) Network throughput and goodput that represent the performance of packet verification by PPV. (iii) Fault localization delay that is to evaluate the efficiency/time of fault localization.

We implemented the PPV scheme described in Section IV and compare it with the state-of-the-art OPT scheme [2]. We use the same encryption algorithm as OPT, \( i.e., \) the AES-CBC-MAC algorithm [28], to perform the MAC operation used in the re/calculation of router marking and update for MVF, and use SHA-3 algorithm [29] to calculate the hash of strings, such as PacketID initialization. For symmetric keys, PPV employs 128-bits keys shared between routers and end-hosts. We evaluate PPV with respect to the performance metrics, described above, for source and path verification.

A. Communication Overhead and Ratio

The communication overhead (CO) is the extra portion of the normal IP packet (\( i.e., \) PPV header and OPT header). From Section IV-A, we know that PPV header is a fixed-length of 64 bytes. However, the length of OPT header varies with forwarding path length \( n \), with the formulation as \( 52 + 16(n + 1) \). Table I shows the CO of PPV and OPT with different path lengths. Specifically, PPV has 43.24% CO of OPT with \( n = 5 \) hops (average AS-level path length [25]), and 23.19% and 16.49% CO of OPT with \( n = 13 \) hops and \( n = 20 \) hops (average and longest router-level path length [26]).

<table>
<thead>
<tr>
<th>Communication overhead</th>
<th>Communication overhead ratio ( (n=13) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>n=5 hops</td>
<td>n=13 hops</td>
</tr>
<tr>
<td>PPV</td>
<td>64 B</td>
</tr>
<tr>
<td>OPT</td>
<td>148 B</td>
</tr>
</tbody>
</table>

We define \( \gamma \) as the proportion between \( CO \) and the entire packet size. For large packets of 1024 bytes, \( \gamma_{PPV} = \frac{64}{1024} = 6.25\% \) and this is independent of path length. However, \( \gamma_{OPT} = \frac{52 + 16(n + 1)}{1024} \), which becomes higher as the path length increases. With \( n = 13 \) hops, \( \gamma_{OPT} = 27.13\% \), worse than \( \gamma_{PPV} = 6.25\% \). More seriously, for \( n = 20 \) hops, OPT incurs 37.11% CO, while PPV only introduces a constant \( \gamma_{PPV} = 6.25\% \). For smaller packets of 256 bytes, \( \gamma_{PPV} = \frac{64}{256} = 25.00\% \) and \( \gamma_{OPT} = \frac{52 + 16(n + 1)}{256} \). Worse still, when the forwarding path length exceeds 12 hops, \( \gamma_{OPT} \) is 100%.
B. Network performance

By analyzing the throughput and goodput, we evaluate the network performance of PPV. To compute goodput, we only consider the packet payload, subtracting IP/TCP header and PPV/OPT header from the entire IP packet. We run the experiment for at least 100 times when routers (destination) forward (receives) $10^6$ packets for computing the throughput and the goodput. As for the basic probability $P_0$, we set it as the value of standard probability ($SP$), i.e., $P_0 = 1/n$. We evaluate the performance of PPV and OPT with different packet sizes from 256 bytes to 1024 bytes by configuring the interface Maximum Transmission Unit (MTU) sizes. Due to $Q_1 = Q_2 = \ldots = Q_n = P_0$, routers at different locations have a same computation overhead. Thus, we can just evaluate the performance of any one router. In this case, different path lengths only bring different $SP$, illustrating we just adjust the value of $SP$ to meet the path length variation.

Router performance. In order to evaluate router overhead, we implement a router prototype for PPV and OPT using the Click Modular Router, a flexible and configurable modular software router [30]. The system running the Click Router is an Ubuntu Linux 12.04, with Intel(R) Core(TM) i5-4590, CPU @ 3.30 GHz, 16.0 GB memory and NIC 1000 Mbps.

With the help of iperf [31], we used a personal computer (Intel(R) Core(TM) i5-4200U, CPU @ 1.60 GHz/2.30 GHz, 12.0 GB memory) as the source to send packets to the NIC (Intel(R) Core(TM) i5-4200U, CPU @ 1.60 GHz/2.30 GHz, 12.0 GB memory) via a NIC at a maximum rate of 1000 Mbps. The Click Router takes packets from NIC and processes them according to PPV or OPT protocol. Then it forwards the packets to the next hop. During this process, we evaluate router overhead by calculating its throughput and goodput for different packet sizes and path lengths. Fig. 6 shows the evaluation for PPV and OPT performance, where we can observe the following results. i) PPV outperforms OPT in terms of throughput and goodput in all but the smallest path lengths. The improvement in PPV is because only two neighbor routers (instead of all in OPT) perform packet marking with MAC operation for each forwarded packet. ii) With increasing path lengths, PPV introduces a better performance, while OPT brings about constant throughput and decreased goodput (described shortly).

From OPT, we can know an OPT router’s operation is independent from the path length, resulting in an almost constant throughput with the fixed packet size. Besides, an OPT router’s goodput gradually decreases due to the increased communication overhead with incremental path lengths. Therefore, its goodput decreases when the path length increases. In contrast, the performance of a PPV router gets better as the path length increases as PPV router $R_i$ has three, mutually exclusive, actions to either: i) insert its marking $M_i$ into $MF_1$ and update $MVF$, with the probability $Q_1 = P_0 = 1/n$; ii) write $MF_2$ as $M_i$ and then update $MVF$, with the probability $Q_1 = P_0 = 1/n$; iii) directly forward the packets to next hop, with the probability $(1 - Q_1 - Q_{i-1}) = (1 - 2/n)$. With increasing path lengths, PPV router tends to carry out action iii), incurring less latency for packets process. Thus, the throughput and goodput of PPV become increasingly higher with larger path lengths. Additionally, PPV’s constant communication overhead also improves the goodput compared to OPT as the throughput increases. With regard to numerical magnitude, PPV actions i) and ii) make PPV router perform a MAC operation twice the number of times as OPT router. In this case, PPV router’s larger overhead in creating PPV header, compared to OPT router, decreases PPV’s benefit when the path length is two hops. However, when the path length is larger than two hops, there is a probability $(1 - 2/n)$ for PPV router to perform action iii), bringing a higher throughput and goodput than OPT.

Destination performance. We implement Click Modular Router to forward received packets to a NIC via a NIC. It supports PPV and OPT running on Windows Server 2008, which has Intel(R) Xeon(R), CPU E5-2620 v3 @ 2.40 GHz (2 processors) and 32.0 G memory and a 1000 Mbps NIC.

When evaluating its performance, we analyze the throughput and goodput for PPV and OPT with different path length and packet size. Furthermore, we evaluate the performance at $P_0 = 0.05$ and $P_0 = 1/n$. From (a) to (d) in Fig. 7, we observe: i) PPV has higher throughput than OPT, because of PPV’s fewer MAC operation (5 times) than OPT’s $((n + 1)$ times) at $P_0 = 1/n$; ii) PPV has higher goodput than OPT due to PPV’s higher throughput and constant communication overhead; iii) When $P_0$ is set to standard probability, i.e., $P_0 = 1/n$, PPV overhead is essentially constant at different packet sizes, illustrating the independence between path length and the performance. In contrast, OPT performs worse as the path length increases, because of the increased computational complexity; iv) With $P_0 = 0.05$, PPV has an improved throughput and goodput, compared to PPV with $P_0 = 1/n$ and is better than...
OPT. In the latter case, an empty marking field can appear on a PPV header with the probability \(1 - nP_0\). Reducing \(P_0\) thus increases the number of empty marking fields, which lowers average time for destination’s verification for one packet.

With the change of \(P_0\), we will next analyze the influence on the performance of PPV. As \(P_0\) is varied, routers and destination will have different throughputs, just as Fig. 7(e) shows. We respectively evaluate the throughputs of PPV and OPT in the Click Module Router and destination host for 256 bytes and 1024 bytes packet sizes. From Fig. 7(e), we see that the increased \(P_0\) can lower the throughput of PPV for different packet sizes, because routers tend to directly forward the packets instead of performing the marking calculation, and at \(D\) it is less overhead to receive an empty marking field in a PPV header. However, the degree of the effect differs between routers and \(D\), because the probability \((1 - nP_0)\) of \(D\)’s receiving empty marking field is more dominant than the probability \((1 - 2P_0)\) that a router directly forwards packets.

C. Fault Localization Time

In this paper, we use fault localization time \(T_f\) to evaluate the time for reconstructing \(\Phi\) and localize the fault. Actually, once \(\Phi\) is constructed, \(D\) can employ Algorithm 2 to localize the fault. \(T_f\) is relevant to both the convergence time \(T_c\), illustrating the required number of packets to reconstruct \(\Phi\), and \(D\)’s performance, which implies the rate of handling packets at \(D\). Combining the analysis of \(T_c\) (Section IV-E) and \(D\)’s performance, we analyze the relationship between \(T_f\) and \(P_0\) at \(D\) host when the path length is almost the average value of the Internet, i.e., \(n = 13\) hops, as Fig. 7(f) shows.

We respectively evaluate \(T_f\) for the packet sizes of 256 bytes, 512 bytes, 768 bytes and 1024 bytes, which is from smaller packets to larger packets. We learn that the smaller packet, e.g., 256 bytes, requires the less \(T_f\) for localizing the fault than the larger packet, e.g., 1024 bytes, because small packets require less time for transmission under the same \(T_c\). When \(P_0\) takes the value of the standard probability, i.e., \(P_0 = 1/13\), 0.50 ms ~ 0.80 ms will be required to locate the fault for different packet sizes. For the packet size of 256, 512, 768 and 1024 bytes, \(P_0 = 1/13\) can bring about the fault localization time of 506.3 µs, 620.3 µs, 706.8 µs and 784.4 µs.

VII. RELATED WORK

Secure routing and forwarding. The security of routing and forwarding is a vital component to ensure the authenticity of the Internet’s routing infrastructure [1] [10] [19] [32] [33] [34] [35]. S-BGP can address the vulnerabilities and security requirements associated with BGP, which consumes a significant amount of router resources [1]. So-BGP cannot only authorize the integrity of packets, but can also mitigate various network-based attacks deriving from malicious insertion or misconfiguration [10]. IRV can validate the reliability of the announced messages at the cost of adding a new infrastructure, bringing about the challenges of maintenance and migration [33]. In contrast, PPV can lessen the burden of packets and improve the performance of intermediate routers using probabilistic packet marking.

Source and path verification. Tiffany Hyun-Jin Kim et al. [2] ensure source and path verification through the Origin and Path Trace (OPT) protocol. OPT can introduce lightweight routers for packet verification. Jad Naous et al. [3] present a Path Verification Mechanism (PVM) to address the verification for both source and path, checking the packet’s supposed path and verifying the packets have passed all previous routing nodes on the path in the correct order. Hao Cai et al. [15] introduce a new protocol, OSP, to avoid the high computational cost.
of cryptographic operations, which uses a set of orthogonal sequences as credentials to a desired level of the accelerated verification in each router.

**Fault localization.** ShortMAC [22] leverages packet counts and content to identify the illegal network links, which achieves high authentication efficiency. DynaFL [36] achieves fault localization in datacenter networks, which has high availability for dynamic traffic patterns. Faultprints [11] can firstly localize the fault at the granularity of AS (inter-domain) level with high performance and lower overhead. However, they all require each router to pay for extra storage overhead.

**IP traceback.** IP traceback solutions can contribute to protecting the destination from DDoS attacks, source spoofing and localizing the faulty source. Pi, short for Path Identifier, employs a new packet marking method to enable the destination to identify each packet traveling along the same path [17]. Abraham Yaar et al. propose the stackPi marking, containing stack-based marking and write-ahead marking, which is a novel packet marking mechanism for DDoS and IP spoofing defense [18]. Stefan Savage et al. introduce a universal traceback method which is based on probability packet marking to help the victim identify the attack traffic [16]. However, in their mechanisms, all entities are supposed credible and secure, which cannot suit the localization of faulty routers.

**VIII. CONCLUSION**

In this paper, we design, analyze, implement and evaluate PPV, a highly-efficient data-plane verification for source authenticity and path compliance, which performs a trade off between security and overhead. PPV employs probabilistic packet marking in entities such that it allows the destination to verify packet source and packet forwarding paths. Meanwhile, PPV can efficiently localize offending entities by reconstructing the actual forwarding paths. We make an security analysis to demonstrate the security of PPV against source spoofing and path inconsistency. We also demonstrate the high-efficiency of PPV by real experiments. PPV outperforms the state-of-the-art schemes in the following aspects. First, PPV effectively decreases the packet forwarding overheads and thus improves the throughput and goodput. Second, PPV is capable of efficient fault localization. We hope that the high-efficiency property of PPV can become a fundamental primitive to build highly secure and readily deployed security protocols.

**ACKNOWLEDGMENTS**

This work was supported by the Natural National Foundation of China (61472212, 61572278, and 61602039), EU Marie Curie Actions CR-OWN (FP7-PEOPLE-2013-IRSES-610524), the R&D Program of Shenzhen (JCY2017030715325932), and National Science Foundation under grant CNS-17-17313.

**REFERENCES**