2025 IEEE 33rd International Conference on Network Protocols (ICNP) | 979-8-3315-0376-5/25/$31.00 ©2025 IEEE | DOI: 10.1109/ICNP65844.2025.11192382

PIPE: Identity-Aware Privacy-Enhanced Source and Path Verification for Strengthened Network Accountability

Jianfeng Guan[1], Kexian Liu[1], Su Yao[2*], Xiaolong Hu[1], Ye Qin[1], Jianli Liu[1], Songtao Fu[3], Xiangyu Gao[3], Ke Xu[3]
[1]State Key Laboratory of Networking and Switching Technology,
Beijing University of Posts and Telecommunications, Beijing, China
[2]Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing, China
[3] Department of Computer Science and Technology, Tsinghua University, Beijing, China
{jfguan, kxliu, hxl814446051, qiny, kuohao233}@bupt.edu.cn, {yaosu, fust18, gao-xy, xuke} @tsinghua.edu.cn

***Abstract*—Network-layer security threats have become increasingly sophisticated, exposing significant vulnerabilities in the current Internet architecture. Despite various proposed solutions, the field faces fundamental challenges in balancing user privacy with network security and achieving practical deployment. This paper presents a novel approach called PIPE (Privacy-preserving Identity and Path Enhancement), which leverages a distributed infrastructure of Key Distribution Servers (KDS) to integrate Decentralized Identity (DID) with source and path verification. Our solution binds user identity, address, path, and data while maintaining privacy through encryption and per-hop address transformation. By embedding DID information in address and implementing encrypted path verification, we achieve enhanced network accountability without compromising privacy. Experimental results demonstrate PIPE's practicality and advantages over existing approaches in terms of deployment flexibility and security guarantees. Our work contributes to the evolution of secure network architectures by balancing accountability requirements with privacy protection while ensuring practical deployability.**

***Index Terms*—accountability, identity authentication, source and path verification, privacy-preserving, path-aware networking**

I. Introduction

The Internet's "narrow waist" design (the unified IP layer) has been crucial for its development, shielding underlying heterogeneity while supporting upper-layer innovations. Over the years, the network has evolved to accommodate various mechanisms and requirements, demonstrating its foundational role [1]. With emerging security threats such as BGP hijacking attacks [2] and AI-powered attacks [3], there is an increasing need for enhanced security and privacy capabilities in network architecture [4]–[7]. Enhancing the network layer's security while preserving ecosystem compatibility thus becomes a critical challenge for Internet evolution [8]–[10].

This work was supported by the National Key R&D Program of China (Grant No. 2022YFB3102304), the National Natural Science Foundation of China (Grants U22B2031, 62472240, 62394322, 62394323, 62225105, 62001057), and the Beijing National Research Center for Information Science and Technology (Grant BNR2025RC01010).
Corresponding authors: Su Yao.

To address these challenges, academia has proposed two main approaches: a clean-slate revolution (fundamental redesign of network architecture) [8], [11]–[13] and an evolutionary enhancement (incremental improvements to existing protocols) [1], [9], [14]–[16]. Notable contributions include DIP's field operations for network function composition [8], Trotsky's configurable L3.5 [1], and ToIP's decentralized identifier (DID) -based trust roots [9]. Meanwhile, solutions like OPT [11], PPV [14], EPIC [12], PAVI [15], and MASK [16] have made valuable explorations in source address and path verification. However, these solutions face a fundamental trade-off: strengthening path verification and source authentication often compromises privacy, while privacy protection can hinder network providers' security control. Despite their technical merits, such inherent limitations have hindered widespread adoption, reflecting several fundamental challenges in current network security:

Inadequate Identity Traceability: Existing solutions (OPT, PPV, EPIC, PAVI, MASK, etc.) only verify packet source addresses or host identifiers without human identity integration. With the rising adoption of W3C's DID standards [17] and users accessing networks through multiple devices, operators struggle to achieve precise user behavior tracing.

Privacy and Security Trade-off: Current solutions face a fundamental challenge in balancing accountability and privacy protection. Existing approaches either excessively expose end users' information (such as including plaintext source addresses and path information in packet extension headers), making users vulnerable to traffic analysis and correlation attacks [18]–[23], or sacrifice network accountability to protect user privacy, preventing network providers from implementing necessary security controls. This inherent tension between accountability and privacy in the general Internet setting makes it challenging for network operators to maintain both effective security management and meaningful privacy protection for users.

Insufficient Deployment Motivation: Most existing solutions not only introduce new identifiers at the technical level,

increasing integration difficulties with existing network infrastructure [15], but more importantly, fail to simultaneously meet the needs of end users and network providers. End users resist adoption due to privacy concerns, while network providers lack deployment motivation due to limited security control capabilities.

These challenges are particularly evident in high-security domains where accountability and privacy must coexist. For instance, in enterprise network security auditing, employees access corporate resources through multiple devices (smartphones, laptops, tablets) across different locations. Traditional IP-based tracking fails when users switch devices, while device-based monitoring cannot distinguish between users sharing the same device. Meanwhile, exposing complete routing paths in packet headers creates privacy risks, as attackers can analyze traffic patterns to infer user locations and behaviors. This scenario illustrates the urgent need for a solution that enables precise user-level tracking while protecting both real identities and routing paths from intermediate network nodes.

A deeper issue is that merely combining existing solutions cannot resolve these issues, as it increases complexity and may introduce new vulnerabilities. The fundamental challenge lies in establishing a secure binding between network packets and users without exposing user identities to intermediate nodes or potential attackers. While the network layer traditionally remains isolated from user-level concerns, emerging security requirements demand a mechanism that can associate packets with users while preserving privacy. This requires a trusted intermediary that can perform identity-to-packet binding while ensuring that neither network intermediaries nor adversaries can reverse this binding to compromise user privacy.

To address these challenges, based on an analysis of end users and network providers requirements, we propose PIPE, which achieves triple integrated verification of identity, source address, and path, injecting new security capabilities into the network layer. The core idea is to establish cross-autonomous systems (ASes) trust through distributed Key Distribution Servers (KDSs), implement verification using cryptographic tags, and integrate DID with network-layer verification mechanisms. Within PIPE, end users obtain comprehensive privacy protection through customizable verification services, while network providers implement verification-as-a-service with minimal overhead, retaining security control capabilities for authorized tracing. PIPE adopts IPv6 extension headers, minimizing modifications to existing infrastructure and providing a feasible path for network layer evolution. The main contributions of this paper are as follows:

- So far as we know, we propose the first mechanism to bind DID with the network layer by embedding DID in IPv6 address interface identifier (IID) and incorporating encrypted path information in extension header. This achieves secure association of identity, address, path, and data[1], enabling network providers to extract user identity information from each packet.
- We design an innovative encryption scheme that achieves identity privacy and flow unlinkability through hop-by-hop transformation of IID, while encrypting path in extension header to ensure each forwarding node only knows its directly adjacent nodes. This approach maintains data confidentiality for unauthorized parties while preserving traceability for authorized parties, all while implementing identity-address-path verification.
- We adopt an IPv6 compliant extension header design, ensuring seamless integration with existing networks. Comparative evaluation with state-of-the-art approaches on the Data Plane Development Kit (DPDK) [24] platform demonstrates balanced performance and enhanced security features. Tests on programmable hardwares (Intel Tofino 2) show only 1$\sim$6 μs additional end-to-end latency and 11$\sim$41 ns additional processing time compared to baseline IPv6, achieving line-rate packet processing.

The rest of the paper is organized as follows: In Section II, we introduce the problem statement. Section III presents the system model of PIPE. Section IV details the proposed PIPE scheme. Section V analyzes the security of PIPE. Section VI presents implementation and evaluation results. In Section VII, we analyze the related work. Finally, we conclude this paper in the last section.

II. Problem Statement

A. Background

IPv6 offers two key advantages for PIPE design. First, its vast address space enables hosts to maintain multiple addresses simultaneously, which facilitates privacy protection through address transformation. Second, its standardized extension header mechanism provides a natural foundation for deploying new security features. To address location privacy concerns when the IPv6 prefix remains unchanged, PIPE implements additional security measures through IID verification mechanism, which we detail in Section IV.

B. Design Goals

Accountability. PIPE ensures accountability through packet authenticity verification, unique user identification, and binding identity to network activities. By embedding DID, we align with self-sovereign identity management, enabling users to maintain consistent identities across devices while controlling their credentials.

Privacy Protection. Our design achieves several security properties: user identity anonymity through anonymized identifiers that prevent attackers from obtaining true identities; source address privacy via hop-by-hop address transformation that hinders traffic analysis; source-destination unlinkability where no observer can link sources to destinations by monitoring traffic flows; and path privacy through encryption where sources and routers access only next-hop addresses, preventing global path disclosure.

Deployability. PIPE prioritizes practical deployment through IPv6 extension headers, requiring no new protocol identifiers or significant infrastructure modifications. Minimal

[1] 'data' refers to packet payload.

performance overhead and backward compatibility enable straightforward adoption in existing networks.

C. Adversary Model

We adopt an adversary model based on Dolev-Yao [25], where the adversary has full control over network channels they can access. This includes the ability to intercept, observe, inject, modify, delete, and replay packets.

Trust Assumptions: Inspired by systems like PISKES [26], our security model assumes each network domain operates a centralized KDS while maintaining inter-domain distributed architecture. We assume KDS core keys are securely managed (potentially within Hardware Security Module (HSM)) and all control plane communications are protected via mutually authenticated encrypted channels.

Our key management employs short-lived symmetric keys for data plane encryption (e.g., k_{sd}, k_i) with frequent rotation, while using long-lived master keys (e.g., K) for control plane operations. This creates a centralized trust point within each domain, aligning with existing AS autonomy structures. For enhanced inter-domain trust, distributed mechanisms like blockchain-based approaches can be leveraged [27], balancing practical deployment with distributed trust principles.

Adversary Capabilities: The adversary possesses complete knowledge of the PIPE protocol specifications but cannot guess or derive the secret cryptographic keys of honest entities unless these are otherwise compromised. The adversary's capabilities are:

1)Interception and Observation: The adversary can read all fields of any packet traversing a compromised link, including the IPv6 header, PIPE extension header (IID, Mark, encrypted PATH, sequence, timestamp), and payload (if not otherwise encrypted).

2)Injection and Forgery: The adversary can craft and inject arbitrary packets. Attempts to forge PIPE headers are contingent upon possessing or deriving the correct cryptographic secrets.

3)Modification: The adversary can intercept, alter (e.g., tamper with IID, Mark, or parts of the encrypted path), and retransmit packets.

4)Deletion/Dropping: The adversary can prevent packets from reaching their destination.

5)Replay: The adversary can record and resend previously intercepted packets.

III. Overview

To achieve user identity management and path verification, we designed the PIPE header, as illustrated in Fig. 1. Considering deployability, this header can be embedded as an IPv6 extension header or as the VALHD header in EPIC [12], enabling compatibility with SCION [28]. In the PIPE header, the $sequence$ field contains the $epoch$ and seq (sequence number) of a packet to prevent replay attacks and enable flow correlation resistance, with detailed mechanisms explained in Section IV.C. The IID field, representing the encrypted form of an anonymized identifier, ensures the unlinkability of PIPE through dynamic updates. $PATH = \{P_0, P_1, ..., P_n\}$ is the encrypted form of the path, allowing each hop to decrypt and retrieve the identifier id_x of the next-hop entity. We assume 32-bit identifiers for routing nodes provide sufficient address space, though source and destination hosts may require longer identifiers given their distinct addressing requirements. The $Mark$ field embeds information related to the *Source* and undergoes per-hop updates along the path, enabling the *Destination* to verify both the path and packet integrity. *Source* authentication verifies the user's registered legitimate DID, while path verification ensures packets are forwarded along predetermined compliant paths.

TABLE I
Notations

Notation	Description
DID	A globally unique, persistent decentralized identifier that represents user identity without requiring centralized registration, typically generated and managed through cryptographic methods.
AID	Anonymous identifier derived from DID
IID	A 64-bit dynamic tag embedded in *Source* address interface identifier, updated at each hop
KDS	Key Distribution Server
id_i	Identifier of the i-th entity
$PATH$	Encrypted path information $\{P_0, P_1, \ldots, P_n\}$ where each P_i is the encrypted identifier of each hop
$Mark$	Cryptographic marker enabling path verification
α_i	Long-lived secret value shared by i-th entity and KDS
ε, β	Intermediate cryptographic parameters for identity and path recovery
$seed$	*Source*-embedded value computed from packet payload and AID
k_i	Symmetric key between the i-th entity and the i-1-th entity
TS_i	Timestamp when the AID is generated
T_i	Timestamp embedded in the PIPE header by i-th entity when processing the packet
CSH	Checkpoint Sequence Hopping array storing expected packet counts per epoch
$A[i:j]$	A subsequence of array A from index i to $j-1$ (0-based indexing, exclusive upper bound)
X'	Primed variables (e.g., id', ε',β',AID') denote values recomputed by downstream nodes to reproduce original calculations, satisfying $X = X'$ under normal operation

Fig. 2 illustrates the detailed processing procedures in the end hosts and routers. PIPE's architecture consists of four key components that interact within a well-defined trust framework:

End Hosts (*Source* and *Destination*): End hosts are the ultimate originators and consumers of network traffic. They establish secure communication channels and embed identity information in packets.

Routers: These network elements perform packet forwarding while participating in the verification process. They hold symmetric keys shared with adjacent routers and perform hop-by-hop tag transformations.

KDS: Trusted entities responsible for key management and distribution across the network. Each AS operates at least one KDS, collectively forming a distributed trust infrastructure.

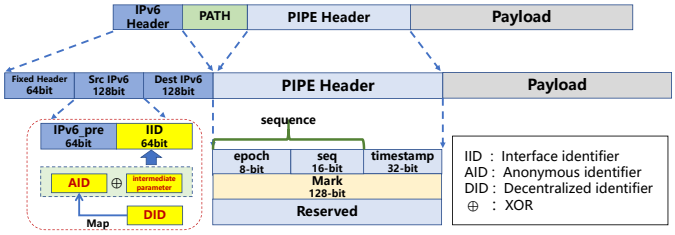


Fig. 1: PIPE Packet

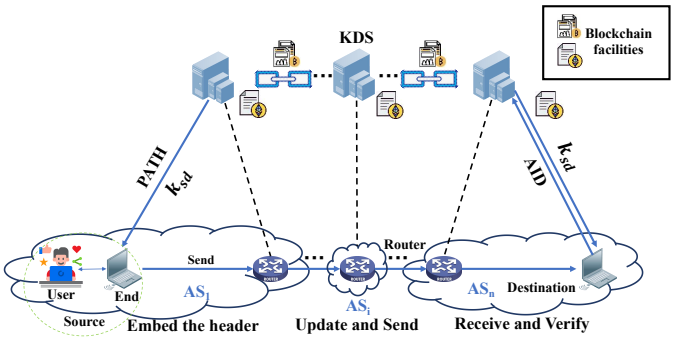


Fig. 2: PIPE Overview

These servers manage the lifecycle of DIDs and their Anonymous Identifier (AID) mappings, while collaboratively maintaining a shared blockchain infrastructure for secure record-keeping.

Blockchain Infrastructure: A distributed ledger maintained by the KDS network that provides immutable identity attestation records. This enables secure verification and authorized updates of identity information (e.g., DID) while ensuring data consistency across the distributed environment.

IV. Details

This section details the PIPE protocol implementation. Table I explains the meaning of important symbols used throughout this paper.

A. Processing Procedure of KDS

Upon receiving a transmission request, the KDS determines the path $(id_S, id_1, id_2, \ldots, id_n, id_D)$ and generates symmetric keys $K_{path} = \{k_{sd}, k_0, k_1, \ldots, k_n\}$, where k_{sd} is shared between *Source* and *Destination*, and k_i between adjacent routers id_i and id_{i-1}. Each key is disclosed only to the respective entities and KDS, while secret value α_i remain exclusive to KDS and router id_i.[2]

To prevent path disclosure, the KDS encrypts path information before transmission. For each entity, it calculates:

$$\beta_i = \text{MAC}_{k_{i-1}}(id_{i-1} \parallel id_i \parallel \alpha_i) \tag{1}$$

$$P_i = \beta_i[0:32] \oplus id_{i+1} \tag{2}$$

$$\beta = \beta_0 \oplus \beta_1 \oplus \ldots \oplus \beta_{n-1} \oplus \beta_n \tag{3}$$

where β_i is the secret value computed by i-th entity for calculating next-hop path information, MAC denotes a message authentication code function keyed with k_{i-1}, id_i is the i-th entity identifier, k_{i-1} is the symmetric key shared between the (i-1) -th and i-th entities, α_i is the secret value shared between the i-th entity and the KDS, and $[0:32]$ indicates truncation to the first 32 bits.[3]

Finally, KDS sends $\{\beta, PATH, k_{sd}\}$ to the *Source*, where $PATH = \{P_0, \ldots, P_n\}$.

B. Distributed Identity Infrastructure

PIPE implements a two-tier identity verification framework. Identity servers generate unique DID from human attributes and record them on a blockchain for immutable verification. For privacy protection, DIDs are transformed into AIDs through:

$$AID = DID \oplus \text{MAC}_K(TS_i)[0:64] \tag{4}$$

where K is the long-lived master key known only by the KDS, and TS_i is the timestamp when the AID is generated. The KDS stores the mapping (AID, TS_i), enabling authorized recovery via $DID = AID \oplus \text{MAC}_K(TS_i)[0:64]$ during traceback while protecting against unauthorized access.

C. Processing Procedure of the Source

1) Identity processing

PIPE incorporates anonymous identifiers into network packets while maintaining security and privacy, embedding identity information within packet headers through encryption and verification.

During transmission, the *Source*'s AID serves as the base anonymous identifier, with each previous hop encrypting the identifier using a symmetric key shared with the next hop:

$$\varepsilon_i = \text{MAC}_{k_{i-1}}(id_{i-1} \parallel id_i \parallel T_i) \tag{5}$$

$$IID_i = \varepsilon_i[0:64] \oplus AID \tag{6}$$

where ε_i is the secret value that can be computed by both i -th entity and (i-1) -th entity, T_i is the timestamp embedded in the PIPE header by i-th entity when processing the packet, which effectively prevents replay attacks and achieves unlinkability.

The resulting IID replaces the rightmost 64 bits of the *Source* address's interface identifier. PIPE thus establishes a secure transformation chain: **DID → AID → IID → Packet Verification**, binding human identity to network activities while preserving privacy.

Verification is achieved by calculating:

$$seed = H(payload) \oplus E_{k_{sd}}(AID) \tag{7}$$

where $payload$ is the packet data and $E_{k_{sd}}()$ represents AES encryption using key k_{sd}. The $seed$ conceals the *Source*'s identity while providing integrity verification.

PIPE implements security through timestamp-based dynamic identity tags and comprehensive integrity protection, combining payload hash with encrypted AID to prevent tampering and traffic analysis. Additionally, dynamic L2 address transformation occurs at each hop using pre-negotiated L2

[2]In our notation, subscript 0 represents the *Source* and subscript $n+1$ the *Destination* (e.g., $id_0 = id_S$ and $id_{n+1} = id_D$).

[3]When $i = 0$, k_{i-1} becomes k_{sd} and id_{i-1} becomes id_D.

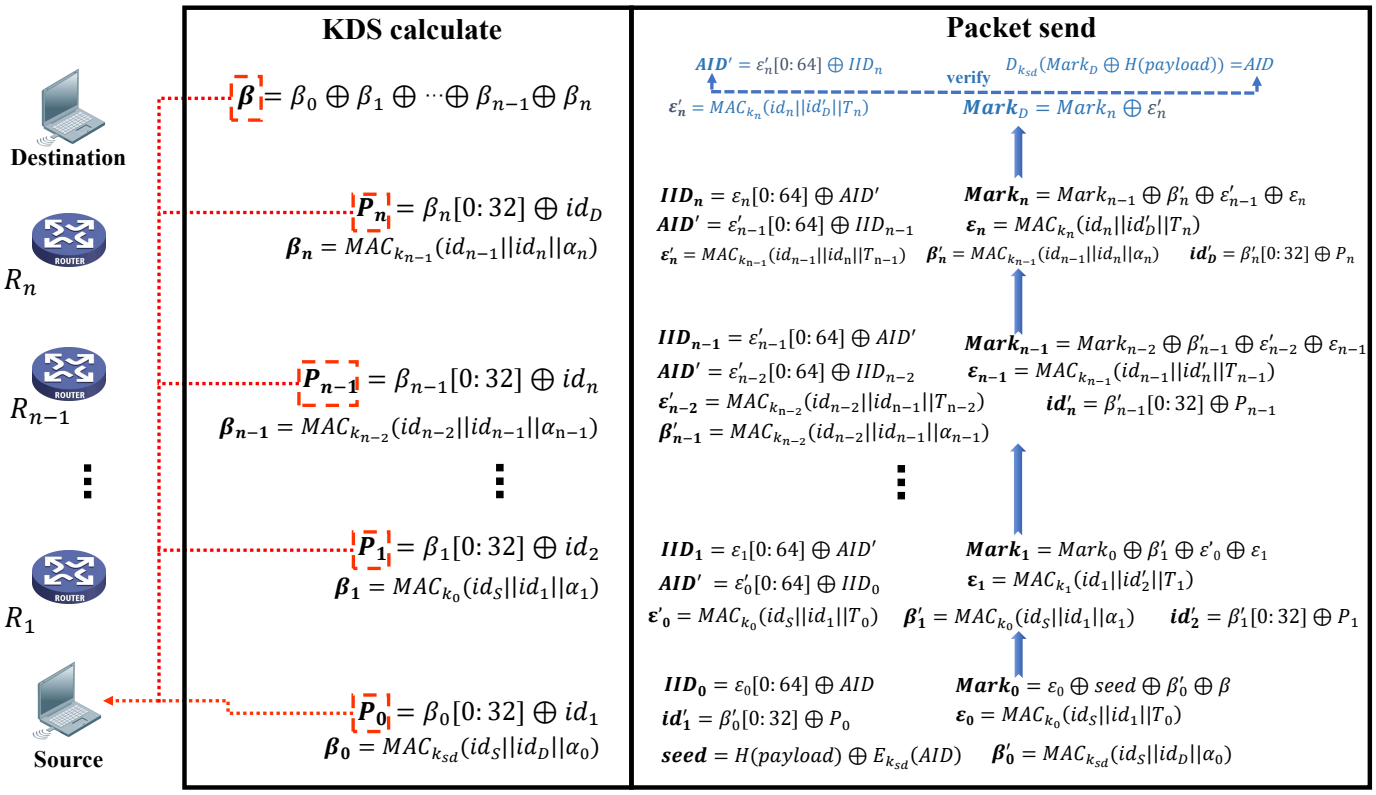


Fig. 3: PIPE Processing Flowchart

address pools, ensuring unlinkability without compromising performance.

2) Path processing

Upon receiving $\{\beta, PATH, k_{sd}\}$, the *Source* derives $seed$ by Eq. (7), which serves as the initial parameter for $Mark_0$. It then calculates β'_0, ε_1, IID_0, id'_1 and $Mark_0$ using Eqs. (1), (5), (6), (8) and (9) to conceal the AID. The key operations are defined by:

$$id_{i+1} = \beta_i[0:32] \oplus P_i \tag{8}$$

$$Mark_0 = \varepsilon_1 \oplus seed \oplus \beta'_0 \oplus \beta \tag{9}$$

The *Source* initializes IID_i and $Mark_i$ to ID_0 and $Mark_0$ respectively, which are dynamically updated by intermediate routers.

Our approach computes an adjustment factor[4] λ to introduce controlled randomness into packet transmission:

$$\lambda = MAC_{k_{sd}}(epoch) \bmod N \tag{10}$$

where N is a configurable parameter and λ ranges from 0 to $N-1$. The adjustment factor determines the exact number of packets in each epoch through $CSH[m] = N + \lambda$, creating epochs containing N to $2N-1$ packets. This forms the CSH array where $CSH[M]$ represents packet counts for M epochs. For example, $CSH[M] = \{4500, 5001, \ldots, CSH[m], \ldots\}$ indicates 4500 packets in epoch 0, 5001 packets in epoch 1, and so forth. Algorithm 1 shows the specific process on the *Source*.

Algorithm 1 Procedure at *Source*

Require: k_{sd}, current epoch m, N, λ

1: Compute $\lambda = MAC_{k_{sd}}(m) \bmod N$
2: Set $CSH[m] = N + \lambda$
3: **for** $seq = 0$ to $CSH[m] - 1$ **do**
4: Set header *epoch* = $(m \bmod 256)$, sequence = seq
5: Calculate IID_1, id'_1, and $Mark_0$ using Eqs. (5), (6), (8) and (9)
6: $\boxed{SourceIP[64:128]} \leftarrow IID_1$, $\boxed{Mark} \leftarrow Mark_0$
7: $\boxed{PATH} \leftarrow \{P_0 ..., P_n\}$
8: Send packet to first hop
9: **end for**

[4]The PIPE header contains two key fields: the *seq* field restarts at 0 for each epoch and increments to $CSH[m] - 1$, while the *epoch* field enables epoch synchronization between *Source* and *Destination* through shared keys and initial epoch values established during session setup. Due to the 8-bit *epoch* field limitation, packets embed $(epoch \bmod 256)$ while the full epoch value is used for λ computation, enabling arbitrary-length epoch sequences without cross-domain coordination.

The adjustment factor λ serves critical security purposes: it varies with each epoch to enable replay attack detection and packet loss identification, while preventing adversaries from predicting traffic flow patterns since the exact packet count remains unpredictable without knowledge of k_{sd}.

D. Processing Procedure of Routers

Upon receiving a message from the previous hop, a router obtains $IID = SourceAddress[64:128]$ and processes the packet following Algorithm 2. Each router id_i in the path computes β_i and the identifier of the next-hop router id'_{i+1} using Eqs. (1) and (8). Subsequently, it calculates $\{\varepsilon'_{i-1}, \varepsilon_i, Mark_i, AID', IID_i\}$ using Eqs. (5), (6) and (11). It is important to note that before calculating IID_i, the router first derives AID' by performing an XOR operation between the IID_{i-1} and ε_{i-1}, which is the inverse operation of Eq. (6).

$$Mark_i = Mark_{i-1} \oplus \beta_i \oplus \varepsilon'_{i-1} \oplus \varepsilon_i \tag{11}$$

The parameter ε serves as an intermediate value at each hop, ensuring the dynamic variation of $Mark$. Through a chain of XOR operations, the $Mark_D$ at the *Destination* can be restored as $Mark_D = seed$ if all operations are done correctly. The router updates the header fields $\{Mark_i, IID_i\}$ and then forwards the packet to the next hop based on id'_{i+1}.

Algorithm 2 Procedure at Router

Require: pkt, k_i, k_{i-1}, id_i, id_{i-1}
1: calculate IID_i, id'_{i+1}, and $Mark_i$ using Eqs. (5), (6), (8) and (9)
2: $\boxed{Mark} \leftarrow Mark_i$ $\boxed{IID} \leftarrow IID_i$
3: send pkt to the next hop

E. Processing Procedure of the Destination

The *Destination* initializes a counter to record the number of successfully verified packets. The array $sCounter[M]$ stores these counters, where $sCounter[m]$ represents the counter for epoch m.

When the *Destination* receives a packet in epoch m, it computes ε'_D using Eq. (5) and derives $Mark_D$ and AID' following:

$$Mark_D = Mark_n \oplus \varepsilon'_D \tag{12}$$

$$AID' = \varepsilon'_D \oplus IID_D \tag{13}$$

For the first packet of a session, the *Destination* queries the KDS for k_{sd} using AID' and caches this key for subsequent packets. Using the obtained k_{sd}, it decrypts $Mark_D$ to calculate $AID = D_{k_{sd}}(Mark_D \oplus H(payload))$ and verifies whether AID matches AID'. If the verification succeeds, the relative counter $sCounter[m]$ is incremented; otherwise, the packet is discarded.

When the number of packets received for epoch m exceeds a predefined threshold, the *Destination* considers epoch $(m-1)$ as completed and performs validation on $sCounter[m-1]$ using Eq. (14):

$$(N+\lambda)\times(1-\sigma_1) < sCounter[m-1] < (N+\lambda)\times(1+\sigma_2) \tag{14}$$

where $(N+\lambda)$ is the expected number of packets for $sCounter[m-1]$ (i.e., $(N+\lambda) = CSH[m-1]$). We set $\sigma_1 = \sigma_2 = 5\%$, meaning the packet loss and replay rates should be below 5%. If the validation fails, it indicates an abnormality in the packet forwarding for epoch $(m-1)$, and the verification for that epoch is deemed unsuccessful.

The *Destination* verifies the *Source* and path information using Algorithm 3.

Algorithm 3 Procedure at Destination

Require: pkt, k_{sd}, id_n, $CSH[M]$, $sCounter[M]$
1: **while** receive a pkt in epoch m **do**
2: calculate $Mark_D$, AID' and Decrypt AID using Eqs. (5), (12) and (13)
3: **if** $AID == AID'$ **then**
4: $sCounter[m]++$
5: **else**
6: filter the packet
7: **end if**
8: **if** the end of epoch $(m-1)$ **then**
9: **if** $sCounter[m-1]$ violate the policy(Eq. (14)) **then**
10: verification failed at epoch $(m-1)$
11: **end if**
12: $sCounter[m-1] \leftarrow \{0\}$
13: **end if**
14: **end while**

V. Analysis

In this section, we select representative solutions from both revolutionary approaches (OPT, EPIC) that propose fundamental architectural changes and evolutionary approaches (PPV, MASK, PAVI) that focus on incremental security enhancements to existing protocols.

A. Security Analysis

Table II presents a comparative overview of security properties among OPT, PPV, EPIC L3, PAVI, MASK, and PIPE, where $\times$ indicates unsatisfied properties and ✓ indicates satisfied ones.

1)Path Verification. Except for PAVI, which focuses primarily on address auditing and tracing, all schemes implement path verification.

2)Identity Verification. OPT, PPV, EPIC L3, and MASK only verify network-level identifiers without tracing to specific users. While both PAVI and PIPE embed user identities within address, PIPE advances with two key improvements: it employs DID-based identities and achieves stronger coupling among user identity, address, path, and data, compared to PAVI's coupling of only user identity, address, and data.

3)Source Verification. While all compared schemes implement *Source* verification, PIPE stands out by validating not only *Source* addresses but also user identities, enabling more granular traceability.

4)No Asymmetric Cryptography. This feature indicates the absence of asymmetric key operations in packet generation and forwarding. Given that asymmetric encryption is computationally intensive and unsuitable for line-rate forwarding, PIPE employs symmetric key encryption, aligning with mainstream

TABLE II
COMPARISON OF SECURITY AND FUNCTIONALITY

	OPT	PPV	EPIC	PAVI	MASK	PIPE
Path Verification	✓	✓	✓	×	✓	✓
Identity Verification	×	×	×	✓	×	✓
Source Verification	✓	✓	✓	✓	✓	✓
No Asymmetric Cryptography	×	×	✓	✓	✓	✓
Sender Anonymity	×	×	×	✓	×	✓
Path Privacy-preserving	×	×	×	×	×	✓
Sender-Receiver Unlinkability	×	×	×	✓	×	✓

approaches like EPIC L3, PAVI, and MASK to improve forwarding efficiency.

5)Privacy Quantification. We provide a rigorous theoretical analysis of PIPE's privacy properties.

Sender Anonymity. In PIPE, sender identity is concealed using Eqs. (4) and (6). Each packet's IID appears pseudo-random due to the MAC-based transformation with unique timestamps.

For an adversary observing m packets from the same sender, the probability of correctly identifying the sender among n potential candidates is bounded by:

$$\Pr[\text{sender identification}] \leq \frac{1}{n} + m \cdot \text{Adv}_{\text{MAC}} \tag{15}$$

where Adv_{MAC} represents the adversary's advantage in distinguishing the MAC from a random function.

Consider a network with $n = 1000$ nodes where an adversary observes $m = 100$ packets from the same sender. Using secure MAC parameters with $\text{Adv}_{\text{MAC}} \approx 2^{-64}$:

$$\begin{aligned}\Pr[\text{sender identification}] &\leq \frac{1}{1000} + 100 \cdot 2^{-64} \\ &\approx 0.001 + 100 \cdot 5.4 \times 10^{-20} \\ &\approx 0.001\end{aligned} \tag{16}$$

Since $m \cdot \text{Adv}_{\text{MAC}} = 100 \cdot 2^{-64} \ll \frac{1}{1000}$, the identification probability approaches the random guessing baseline $\frac{1}{n} = 0.001$. This means the adversary gains no advantage over random guessing, demonstrating strong sender anonymity.

Path Indistinguishability (PI). In PIPE, path information is encoded as described in Eqs. (1) and (2). For an adversary controlling k out of n nodes on a path, the probability of successfully reconstructing the complete path depends on how many node identifiers remain unknown. Let C be the set of compromised nodes and $A(C)$ be their adjacent nodes that the adversary can observe. The probability of complete path reconstruction is bounded by:

$$\Pr[\text{path reconstruction}] \leq \left(\frac{1}{N_{id}}\right)^{n-|C \cup A(C)|} \tag{17}$$

where $N_{id} = 2^{32}$ represents the total number of possible 32-bit node identifier values. In the worst case, each compromised node reveals up to 3 positions (itself and two neighbors), so $|C \cup A(C)| \leq 3k$, giving:

$$\Pr[\text{path reconstruction}] \leq \left(\frac{1}{2^{32}}\right)^{n-3k} \quad \text{for } n > 3k \tag{18}$$

Since $\frac{1}{2^{32}} \approx 2.3 \times 10^{-10}$ is extremely small and raised to the power of $(n - 3k)$, the probability of complete path reconstruction becomes exponentially negligible, making path recovery computationally challenging for adversaries.

Sender-Receiver Unlinkability. PIPE transforms each packet's *Source* identifier at every hop as described in Eqs. (5) and (6). For packets p_1 and p_2 with identifiers IID_i^1 and IID_i^2, an adversary attempting to link senders with receivers by computing:

$$\text{IID}_i^1 \oplus \text{IID}_i^2 = \varepsilon_i^1[0:64] \oplus \varepsilon_i^2[0:64] \tag{19}$$

gains no useful information, as the result appears indistinguishable from random values. The probability of successfully identifying that two packets belong to the same sender-receiver communication is bounded by:

$$\Pr[\text{successful linkage}] \leq \frac{1}{2^{64}} + \text{Adv}_{\text{MAC}} \tag{20}$$

For secure MAC parameters, this probability approaches the random guessing baseline of $2^{-64} \approx 5.4 \times 10^{-20}$, making sender-receiver linkage computationally infeasible even in large-scale networks.

6)Attack Resistance: PIPE's cryptographic design provides comprehensive protection against the adversary capabilities defined in SectionII.C. Against *interception and observation*, PIPE's dynamic per-hop IID transformation prevents adversaries from correlating packets or identifying senders, while encrypted path information conceals routing details. For *injection and forgery* attempts, without knowledge of the secret keys k_{i-1} and α_i, adversaries cannot generate valid β_i values required for the *Mark* field, resulting in detection at the next honest hop or *Destination*. Regarding *modification attacks*, any tampering with IID, Mark, or path information invalidates the cryptographic verification chain, as the modified *Mark* field will not satisfy $Mark_D = seed$ at the *Destination*. Against *packet dropping*, while PIPE cannot prevent this physical attack, it enables detection through the CSH mechanism, which verifies the expected packet count per epoch. Finally, for *replay attacks*, PIPE implements both sequence numbers and timestamps within the ε_i computation, ensuring that replayed packets from previous epochs or with duplicate sequence numbers are detected and discarded by the validation process at the *Destination*.

B. Performance Analysis

TABLE III
THE OVERHEAD COMPARE WITH OTHER SCHEMES

	Computation Overhead(hop)	Communication Overhead		
		Size(B)	n=5(B)	n=10(B)
OPT	n	16n+68	148	228
PPV	2	64	64	64
EPIC	n	5n+24	49	74
PAVI	2	36	36	36
MASK	n	24	24	24
PIPE	n	16	16	16

Note: n represents the path length.

As shown in Table III, we compare the computational and communication overhead of PIPE with OPT, EPIC L3, PPV, PAVI, and MASK, as these protocols provide similar security guarantees.

- Computational Overhead. Cryptographic operations constitute the majority of computational costs across these protocols. MASK effectively optimizes per-packet encryption by requiring MAC calculations only once per hop along the path. While OPT and EPIC L3 perform operations at every hop, and PPV operates between adjacent hops, PAVI limits computations to egress or ingress border routers between different domains. PIPE implements hop-by-hop verification, achieving enhanced security performance.
- Communication Overhead. The protocols show distinct characteristics in their header size requirements. MASK, PAVI, PPV, and PIPE maintain constant additional header sizes, making them more predictable in terms of overhead. In contrast, OPT and EPIC L3 have variable header sizes that depend on the path length. We demonstrate the communication overhead using 5-hop and 10-hop scenarios, as these represent realistic Internet routing scenarios - the average path length in today's Internet is less than 5 AS-level hops, with the vast majority of Internet AS-level paths not exceeding 10 hops.

It is noteworthy that the EPIC team later introduced the COLIBRI approach, which offers some exciting optimizations [29]. Unlike EPIC, COLIBRI integrates the components S_i and V_i, resulting in reduced communication overhead. However, the computational cost remains the same. The primary aim of COLIBRI is to establish a collaborative, lightweight inter-domain bandwidth reservation infrastructure, overcoming the limitations of previous reservation systems while ensuring worst-case minimum bandwidth guarantees. Since the focus of our work differs from this approach, we do not delve into further discussion of COLIBRI in this paper.

VI. Evaluation

To demonstrate PIPE's feasibility, we conducted comprehensive evaluations on both software platforms (DPDK) and hardware platforms (P4 programmable switches) under diverse network conditions.

A. Software Platform Performance Analysis

We designed a multi-hop network topology closely simulating real-world Internet conditions. Our hybrid testbed combines VMware Workstation virtual machines serving as end hosts with a physical server cluster implementing the routing infrastructure. This arrangement mirrors actual deployment scenarios where end users connect through diverse client devices while traffic traverses dedicated network equipment. End hosts running Ubuntu handle packet transmission and reception, while the physical router nodes employ DPDK as the data plane to ensure high throughput and precise packet processing in real network conditions. As shown in Fig. 4, our experimental testbed employs a complex topology with heterogeneous link capacities to simulate realistic Internet conditions.

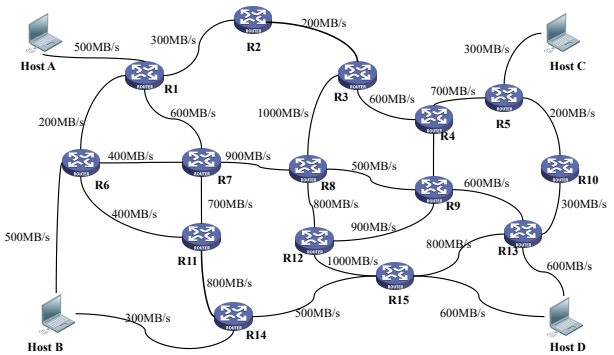
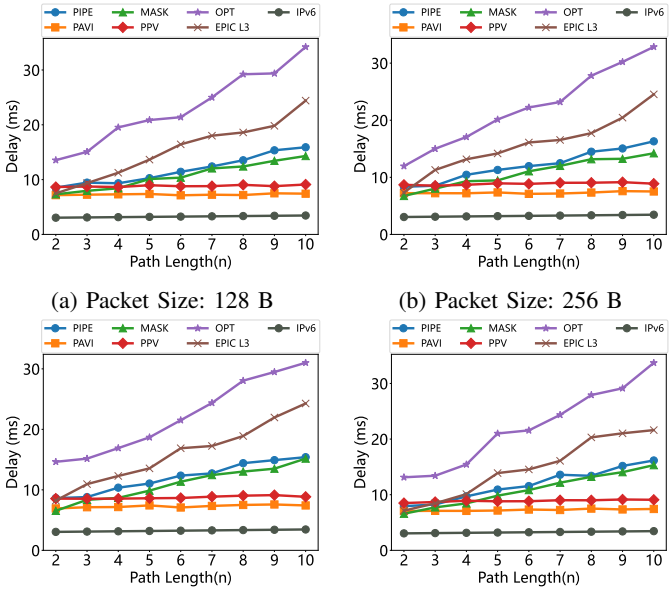


Fig. 4: Experimental network topology with diverse link capacities used in our evaluation.

The physical router cluster utilizes Intel® Xeon® Gold 6133 CPUs @ 2.50GHz and 4 GB RAM per node, providing realistic processing capabilities comparable to production network equipment. Our testbed includes 15 network nodes with varying link capacities (200Mbps∼1000Mbps), operating under different congestion levels . Each experiment was repeated 100 times to ensure statistical significance, with results showing standard deviations below 5% across all metrics.

For fair comparison, we implemented the core processes of representative security mechanisms (PAVI, MASK, PPV, OPT, and EPIC) within identical physical environments. The following results represent averaged measurements across all test iterations.



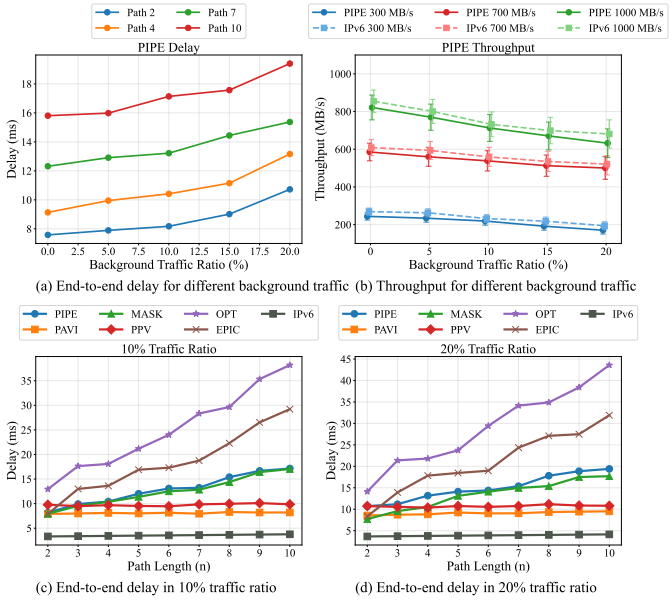
(a) Packet Size: 128 B (b) Packet Size: 256 B

(c) Packet Size: 512 B (d) Packet Size: 1024 B

Fig. 5: The end-to-end delay with different packet size and path length.

To evaluate end-to-end performance, we focused on the delay incurred during packet forwarding. The key components of this delay include the MAC computation overhead at each

node and the latency associated with traditional packet forwarding and transmission. Fig. 5 illustrates the impact of various scenarios on overall network performance, measured as end-to-end delay for different packet load sizes. In a traditional network, the IPv6 packet forwarding and transmission delay is approximately 3 ms. Experimental results indicate that PIPE's overhead lies between MASK and EPIC L3, with a delay averaging around 16 ms for a path length of 10. In the worst-case scenario, where nodes cannot offload MAC operations to higher-performance devices, schemes like PIPE, OPT, EPIC L3, and MASK exhibit increased latency as the path length grows, due to the intensive MAC operations performed at each hop. In contrast, PPV and PAVI centralize computational complexity at a limited number of router nodes, significantly reducing overall transmission overhead and maintaining stable latency across varying path lengths.



(a) End-to-end delay for different background traffic (b) Throughput for different background traffic

(c) End-to-end delay in 10% traffic ratio (d) End-to-end delay in 20% traffic ratio

Fig. 6: Performance analysis of PIPE: (a) Delay measurements across different path lengths with varying link utilization rates; (b) Throughput performance under different bandwidth conditions; (c) and (d) Comparative delay analysis between PIPE and existing approaches at 10% and 20% link utilization respectively.

Fig. 6 demonstrates PIPE's performance compared to existing IPv6-based approaches under various network conditions and path lengths. Background Traffic Ratio indicates the percentage of available bandwidth consumed by background traffic flows, with values ranging from 0% to 20%. The data shows that PIPE achieves competitive performance, with throughput only 3.85% lower than IPv6-based approaches at 1000 Mbps with no background traffic, and this gap remains modest (7.15% difference) even under heavy congestion with 20% background traffic.

The computational overhead at the end host is primarily determined by the device's MAC processing capabilities. In PIPE, the end host needs to request path information and keys from the KDS, which could potentially introduce additional latency. However, as our KDSs are high-performance distributed servers, the latency incurred by these requests is negligible. Furthermore, to minimize any potential delays, end host can cache the received paths and keys for subsequent packets, and a local cache within the domain can be used to store path information.

Incremental Deployment: PIPE maintains full compatibility with existing IPv6 infrastructure through careful protocol design. By implementing verification as a new option within the standard IPv6 Hop-by-Hop Options header, PIPE ensures that non-PIPE-aware routers simply forward packets normally according to IPv6 standard specifications without affecting regular traffic processing. As mentioned in [30], only PIPE-enabled routers interpret and process the verification option, performing security checks while maintaining the packet's original forwarding behavior. This approach allows gradual deployment across networks without requiring simultaneous infrastructure upgrades, as verification only occurs where supported while regular forwarding functions remain unaffected elsewhere.

B. Line-rate Processing Feasibility

We conducted experiments on hardware P4 switches (Intel Tofino 2) to evaluate PIPE's deployability and feasibility. P4's programmability and hardware-level performance capabilities provide an ideal platform for implementing and testing new security protocols while maintaining line-rate packet processing.

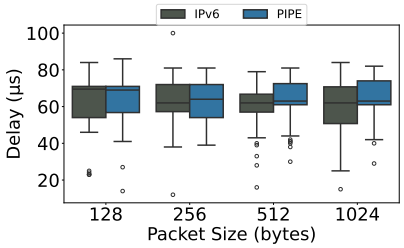


Fig. 7: End-to-end latency.

Fig. 7 show that PIPE maintains competitive performance across different packet sizes (128B to 1024B). With packet size of 128B, PIPE introduces only an average 2μs additional latency compared to IPv6 (average 63.88μs vs 61.86μs). This minimal overhead remains consistent as packet size increases, with the difference staying within an average 1∼6 μs across all test cases, demonstrating PIPE's efficiency in processing packets of varying sizes.

TABLE IV
P4(ROUTER) PROCESSING TIME

Packet Size/B	128	256	512	1024
IPv6 (ns)	379.23	389.06	391.94	690.32
PIPE (ns)	394.64	400.97	402.73	731.27

Table IV demonstrates PIPE's minimal processing overhead, introducing only 11∼41 ns additional processing time compared to baseline IPv6. This combines both ideas - showing

that the table demonstrates PIPE's minimal impact while specifying the exact range of additional processing time.

Despite providing enhanced security, PIPE utilizes resources efficiently—using only 5 of 20 P4 switch stages (25% vs IPv6's 5%), while consuming less Static Random Access Memory (SRAM) (0.5% vs 2.5%) and requiring minimal Ternary Content-Addressable Memory (TCAM) (2.5% vs 0%).

In summary, PIPE achieves line-rate forwarding with negligible performance impact: 1∼6 μs additional end-to-end latency while maintaining consistent performance across packet sizes (128B∼1024B). These results, coupled with PIPE's efficient hardware utilization, enable 12.8 Tbps throughput on Tofino 2 switches, confirming its viability for high-speed networks.

VII. Related Work

Accountability and Privacy. Persona [31], APIP [32], APNA [33], and PAVI [15] are protocols that balance accountability and privacy in the Next-Generation Internet (NGI). Persona enhances anonymity by encrypting source IPs, but struggles with accountability due to route asymmetry and lack of integrity protection. APIP splits source addresses into accountability and return addresses, enabling privacy masking, but faces performance issues with PKI certificate verification and delegated communication. APNA uses ephemeral identifiers (EphIDs) to maintain privacy but is vulnerable to man-in-the-middle attacks and deployment challenges. PAVI and PIPE offer a lightweight, deployable solution without major infrastructure changes, ensuring privacy while maintaining accountability. Compared to other accountability protocols (e.g., AaaS [34], AIP [35], BAFI [36], IPA [37]), which focus on accountability at the cost of privacy, PIPE ensures both privacy and accountability.

Source and Path Verification. Source and path verification has been a key area of study in network security [38]–[41]. Various mechanisms have been proposed to achieve this goal. For instance, Pi [38] and SNAPP [42] use path identifiers to detect DDoS attacks and ensure consistent forwarding paths, but these identifiers are susceptible to forging by attackers. ICING [43] tackles this by computing a MAC for each downstream router, although this method requires multiple MAC operations per packet. In contrast, OPT [11] uses DRKey to dynamically share keys between hosts and routers, while RFL [44] adapts the verification process to unreliable communication channels. EPIC [12] improves the security and efficiency of path verification by introducing a hierarchical security model (L0-L4), which addresses different levels of security needs. For multipath routing, Atlas [45] extends the verification process, while Atomos [40] employs noncommutative homomorphic encryption to provide constant-size proofs, though this comes at the cost of high computational overhead. To minimize overhead, PPV [14] uses probabilistic marking, but this does not allow for source verification within routers. OSV [46] offers a more efficient system by using orthogonal sequences, yet it introduces complexity in the control plane, which limits its scalability and deployability. MASK [16] offloads most verification overhead to the end host and KDS, providing a more practical and deployable solution for source and path verification. SwiftParade [47] introduces composite validation that simultaneously validates multiple-path packet groups using noncommutative homomorphic asymmetric encryption, reducing per-packet overhead while maintaining security through theoretical analysis. Symphony [48] proposes a novel path verification approach using shared validation codes across multiple packets, achieving significant overhead reduction through group-wise validation, adopting a different methodology from per-packet verification schemes. FABRID [49] addresses privacy concerns in network routing by protecting end host policy preferences rather than completely hiding the path sequence. This approach demonstrates the importance of privacy protection in routing systems. However, most of these solutions neglect the privacy protection of the source and path [23], which could lead to various security issues. To address these limitations, PIPE enables source and path privacy while maintaining precise traceability capabilities.

Limitations. PIPE performs Source and Path Verification primarily at destinations rather than intermediate routers. This design choice stems from our path privacy protection approach where cryptographic keys are negotiated between adjacent hop pairs rather than source-to-every-hop (unlike MASK/OPT/EPIC). This distributed key management prevents global path information leakage and enables localized key updates, but creates the limitation that intermediate routers cannot perform real-time attack detection based on source identity verification.

VIII. Conclusion

This paper presents PIPE, a novel approach that integrates DID with source and path verification through a distributed infrastructure. PIPE addresses three critical challenges in current path-aware networking: insufficient accountability, privacy vulnerability, and deployment difficulty. By binding user identity, address, path, and data while maintaining privacy through encryption and per-hop address transformation, PIPE demonstrates a practical compromise where external privacy protection coexists with internal auditability via transformed identifiers. Experimental results show that PIPE achieves its security objectives with minimal overhead compared to state-of-the-art approaches, while deployment on Intel Tofino 2 ensures compatibility with existing infrastructures. The design approach demonstrated provides a foundation for developing next-generation network security protocols that balance accountability, privacy, and deployability.

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions that helped improve this paper. **Jianfeng Guan** and **Kexian Liu** contributed equally to this work and should be considered co-first authors.

REFERENCES

[1] J. McCauley, Y. Harchol, A. Panda, B. Raghavan, and S. Shenker, "Enabling a permanent revolution in Internet architecture," in *Proc. ACM SIGCOMM'19*, pp. 1–14, 2019.

[2] P. Sermpezis, V. Kotronis, P. Gigis, X. Dimitropoulos, D. Cicalese, A. King, and A. Dainotti, "ARTEMIS: Neutralizing BGP hijacking within a minute," *IEEE/ACM Trans. Netw.*, vol. 26, no. 6, pp. 2471–2486, 2018.

[3] N. Kaloudi and J. Li, "The AI-based cyber threat landscape: A survey," *ACM Comput. Surv.*, vol. 53, no. 1, pp. 1–34, 2020.

[4] J. Cao, M. Xu, Q. Li, K. Sun, and Y. Yang, "The loft attack: Overflowing SDN flow tables at a low rate," *IEEE/ACM Trans. Netw.*, vol. 31, no. 3, pp. 1416–1431, 2022.

[5] C. Fu, Q. Li, M. Shen, and K. Xu, "Realtime robust malicious traffic detection via frequency domain analysis," in *Proc. ACM CCS'21*, pp. 3431–3446, 2021.

[6] X. Feng, Q. Li, K. Sun, C. Fu, and K. Xu, "Off-path TCP hijacking attacks via the side channel of downgraded IPID," *IEEE/ACM Trans. Netw.*, vol. 30, no. 1, pp. 409–422, 2021.

[7] K. Liu, J. Guan, S. Yao, L. Wang, and H. Zhang, "DKGAuth: Blockchain-assisted distributed key generation and authentication for cross-domain intelligent IoT," *IEEE Internet Things J.*, vol. 11, no. 15, pp. 25663–25673, 2024.

[8] Z. Wang, Z. Liu, X. Wang, S. Fu, and K. Xu, "DIP: unifying network layer innovations using shared L3 core functions," in *Proc. 21st ACM Workshop Hot Topics Netw.*, pp. 60–67, 2022.

[9] M. Davie, D. Gisolfi, D. Hardman, J. Jordan, D. O'Donnell, and D. Reed, "The trust over IP stack," *IEEE Commun. Stand. Mag.*, vol. 3, no. 4, pp. 46–51, 2019.

[10] D. Clark, C. Testart, M. Luckie, and K. Claffy, "A path forward: improving Internet routing security by enabling zones of trust," *J. Cybersecur.*, vol. 10, no. 1, pp. 1–17, 2024.

[11] T. H.-J. Kim, C. Basescu, L. Jia, S. B. Lee, Y.-C. Hu, and A. Perrig, "Lightweight source authentication and path validation," in *Proc. ACM SIGCOMM'14*, pp. 271–282, 2014.

[12] 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 *Proc. 29th USENIX Security Symp.*, pp. 541–558, 2020.

[13] L. Chuat, M. Legner, D. Basin, D. Hausheer, S. Hitz, P. Müller, and A. Perrig, "The complete guide to SCION," *Information Security and Cryptography*, 2022.

[14] B. Wu, K. Xu, Q. Li, Z. Liu, Y.-C. Hu, M. J. Reed, M. Shen, and F. Yang, "Enabling efficient source and path verification via probabilistic packet marking," in *Proc. IEEE/ACM IWQoS'18*, pp. 1–10, 2018.

[15] L. He, G. Ren, Y. Liu, and J. Yang, "PAVI: Bootstrapping accountability and privacy to IPv6 Internet," *IEEE/ACM Trans. Netw.*, vol. 29, no. 2, pp. 695–708, 2021.

[16] S. Fu, Q. Li, M. Zhu, X. Wang, S. Yao, Y. Guo, X. Du, and K. Xu, "MASK: practical source and path verification based on Multi-AS-Key," *IEEE/ACM Trans. Netw.*, vol. 31, no. 4, pp. 1478–1493, 2022.

[17] M. Sporny, A. Guy, M. Sabadello, and D. Reed, "Decentralized identifiers (DIDs) v1.0." https://www.w3.org/TR/did-core/, 2022.

[18] W. Tan, K. Xu, and D. Wang, "An anti-tracking source-location privacy protection protocol in WSNs based on path extension," *IEEE Internet Things J.*, vol. 1, no. 5, pp. 461–471, 2014.

[19] A. A. Abba Ari, O. K. Ngangmo, C. Titouna, O. Thiare, A. Mohamadou, and A. M. Gueroui, "Enabling privacy and security in cloud of things: Architecture, applications, security & privacy challenges," *Appl. Comput. Inf.*, vol. 20, no. 1/2, pp. 119–141, 2024.

[20] X. Feng, Q. Li, K. Sun, K. Xu, B. Liu, X. Zheng, Q. Yang, H. Duan, and Z. Qian, "PMTUD is not panacea: Revisiting IP fragmentation attacks against TCP," in *Proc. NDSS'22.*, pp. 1–18, 2022.

[21] E. Papadogiannaki and S. Ioannidis, "A survey on encrypted network traffic analysis applications, techniques, and countermeasures," *ACM Comput. Surv.*, vol. 54, no. 6, pp. 1–35, 2021.

[22] T. Cui, G. Gou, G. Xiong, Z. Li, M. Cui, and C. Liu, "SiamHAN: IPv6 address correlation attacks on TLS encrypted traffic via siamese heterogeneous graph attention network," in *Proc. 30th USENIX Security Symp.*, pp. 4329–4346, 2021.

[23] D. Chagnon, K. Thiry-Atighehchi, and G. Chalhoub, "A survey on path validation: Towards digital sovereignty," *Comput. Netw.*, vol. 256, pp. 1–23, 2025.

[24] DPDK, "Data plane development kit." http://dpdk.org/, 2019.

[25] D. Dolev and A. Yao, "On the security of public key protocols," *IEEE Trans. Inf. Theory*, vol. 29, no. 2, pp. 198–208, 1983.

[26] B. Rothenberger, D. Roos, M. Legner, and A. Perrig, "PISKES: Pragmatic Internet-scale key-establishment system," in *Proc. ACM ASIACCS'20*, pp. 73–86, 2020.

[27] X. Wang, K. Xu, Y. Guo, H. Wang, S. Fu, Q. Li, B. Wu, and J. Wu, "Toward practical inter-domain source address validation," *IEEE/ACM Trans. Netw.*, vol. 32, no. 4, pp. 3126–3141, 2024.

[28] J. de Ruiter and C. Schutijser, "Next-generation Internet at terabit speed: SCION in P4," in *Proc. ACM CoNEXT'21*, pp. 119–125, 2021.

[29] G. Giuliari, D. Roos, M. Wyss, J. A. García-Pardo, M. Legner, and A. Perrig, "Colibri: a cooperative lightweight inter-domain bandwidth-reservation infrastructure," in *Proc. ACM CoNEXT'21.*, pp. 104–118, 2021.

[30] J. Guan, S. Yao, K. Liu, X. Hu, and J. Liu, "Terminal identity authentication based on address label," Internet-Draft draft-guan-6man-ipv6-id-authentication-03, Internet Engineering Task Force, 2025.

[31] Y. Mallios, S. Modi, A. Agarwala, and C. Johns, "Persona: Network layer anonymity and accountability for next generation Internet," in *Proc. 24th IFIP TC 11 Int. Inf. Security Conf.*, pp. 410–420, 2009.

[32] D. Naylor, M. K. Mukerjee, and P. Steenkiste, "Balancing accountability and privacy in the network," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 4, pp. 75–86, 2014.

[33] T. Lee, C. Pappas, D. Barrera, P. Szalachowski, and A. Perrig, "Source accountability with domain-brokered privacy," in *Proc. ACM CoNEXT'16.*, pp. 345–358, 2016.

[34] A. Bender, N. Spring, D. Levin, and B. Bhattacharjee, "Accountability as a service," *SRUTI*, vol. 7, pp. 1–6, 2007.

[35] D. G. Andersen, H. Balakrishnan, N. Feamster, T. Koponen, D. Moon, and S. Shenker, "Accountable Internet protocol (AIP)," in *Proc. ACM SIGCOMM'08*, pp. 339–350, 2008.

[36] J. Mirkovic and P. Reiher, "Building accountability into the future Internet," in *Proc. 4th Workshop Secure Netw. Protocols*, pp. 45–51, 2008.

[37] A. Li, X. Liu, and X. Yang, "Bootstrapping accountability in the Internet we have," in *Proc. USENIX NSDI'11*, 2011.

[38] A. Yaar, A. Perrig, and D. Song, "Pi: A path identification mechanism to defend against DDoS attacks," in *Proc. IEEE Symp. SP'03*, pp. 93–107, 2003.

[39] Q. Li, Y. Liu, Z. Liu, P. Zhang, and C. Pang, "Efficient forwarding anomaly detection in software-defined networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 11, pp. 2676–2690, 2021.

[40] A. He, K. Bu, Y. Li, E. Chida, Q. Gu, and K. Ren, "Atomos: Constant-size path validation proof," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 3832–3847, 2020.

[41] Q. Li, Y. Chen, P. P. Lee, M. Xu, and K. Ren, "Security policy violations in SDN data plane," *IEEE/ACM Trans. Netw.*, vol. 26, no. 4, pp. 1715–1727, 2018.

[42] B. Parno, A. Perrig, and D. Andersen, "SNAPP: Stateless network-authenticated path pinning," in *Proc. ACM ASIACCS'08*, pp. 168–178, 2008.

[43] J. Naous, M. Walfish, A. Nicolosi, D. Mazieres, M. Miller, and A. Seehra, "Verifying and enforcing network paths with ICING," in *Proc. ACM CoNEXT'11.*, pp. 1–12, 2011.

[44] 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. Netw.*, vol. 158, pp. 158–174, 2019.

[45] L. Ma, K. Bu, N. Wu, T. Luo, and K. Ren, "Atlas: A first step toward multipath validation," *Comput. Netw.*, vol. 173, pp. 1–15, 2020.

[46] H. Cai and T. Wolf, "Source authentication and path validation with orthogonal network capabilities," in *Proc. IEEE NFOCOM WKSHPS'15*, pp. 111–112, 2015.

[47] A. He, K. Bu, J. Huang, Y. Pang, Q. Gu, and K. Ren, "SwiftParade: Anti-burst multipath validation," *IEEE Trans. Dependable Secure Comput.*, vol. 21, no. 4, pp. 2720–2734, 2024.

[48] A. He, J. Fu, K. Bu, R. Zhou, C. Miao, and K. Ren, "Symphony: Path validation at scale," in *Proc. NDSS'24.*, pp. 1–18, 2024.

[49] C. Krähenbühl, M. Wyss, D. Basin, V. Lenders, A. Perrig, and M. Strohmeier, "FABRID: Flexible Attestation-Based routing for Inter-Domain networks," in *Proc. 32nd USENIX Security Symp.*, pp. 5755–5772, 2023.