I Know If the Journey Changes: Flexible Source and Path Validation

Fan Yang*, Ke Xu*†‡, Qi Li§, Rongxing Lu¶, Bo Wu*, Tong Zhang‖, Yi Zhao* and Meng Shen**‡
*Department of Computer Science and Technology, Tsinghua University, Beijing, China †BNRist, Beijing, China
‡Peng Cheng Laboratory (PCL), Shenzhen, China §Graduate School at Shenzhen, Tsinghua University, Shenzhen, China
¶Faculty of Computer Science, University of New Brunswick, Canada
‖College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China
**School of Computer Science, Beijing Institute of Technology, Beijing, China
{y-f14@tsinghua.org.cn, xuke@tsinghua.edu.cn, qi.li@sz.tsinghua.edu.cn, rlu1@unb.ca, wub14@mails.tsinghua.edu.cn,
zhangt@nuaa.edu.cn, zhaoyi16@mails.tsinghua.edu.cn, shenmeng@bit.edu.cn}

***Abstract*—No matter from the perspective of detection or defense, source and path validations are fundamentally primitive in constructing security mechanisms to greatly enhance network immunity in the face of malicious attacks, such as injection, traffic hijacking and hidden threats. However, existing works for source and path verification still impose a non-trivial operational overhead and lack adjustment capability for path dynamic changes. In this paper, we propose a flexible and convenient source and path validation protocol called PSVM, which uses an authentication structure PIC composed of ordered pieces to carry out packet verification. Specifically, in the basic PSVM protocol, PIC (related to cryptographic computation) in the packet header does not require any update during packet verification, which thus enables a lower processing overhead in routers. To cope with the challenge of path policy changes in the running protocol, the dynamic PSVM protocol supports controllable adjustment and migration, especially in the case of avoiding a malicious node or region. Our evaluation of a prototype experiment on Click demonstrates that the verification efficiency of PSVM is barely influenced by payload size or path length. Compared to the baseline of normal IP routing, the throughput reduction ratio of the basic PSVM is about 13%, which is much better than 28% of existing best solution Origin and Path Trace (OPT). In addition, for a 35-hop path with 30 pieces of PIC needed to be adjusted in dynamic PSVM, the throughput reduction ratio of routing cross node performing the adjustment operation after normal verification is only 2.4%.**

***Index Terms*—source and path validation, PSVM, dynamic verification**

I. Introduction

It is inevitable that current networks are so popular that we cannot image our today's lives without them [1]. Nevertheless, it is still very challenging to make sure a given policy has been properly implemented in current networks, even for some basic path policies. In particular, since network users cannot confirm the source authenticity of data, and network operators also cannot guarantee that the user packets are not detoured in transmission, numerous network attack surfaces are opened up today [2]–[4]. For example, an attacker may try to flood arbitrary packets from multiple spoofed sources to waste downstream resources. Traffic hijacking happens frequently [5] where packets may go through an eavesdropping node, resulting in a potential leakage of sensitive information [6]. It is easy to tamper packet contents on the path and insert other loads such as malicious code.

978-1-7281-6887-6/20/$31.00 © 2020 IEEE

Although a great deal of attention has been paid to source authentication [7], [8], the verification of a packet's actual path has been neglected by comparison. Some existing approaches for IP path tracking fail to solve the above problems, being unable to identify path deviation [9], [10]. Recently, there are several proposals addressing both source and path validation that fill the void, such as ICING [3] and OPT [2]. However, for the targeting environment which is adversarial, high-speed, and variable-path, their protocols still have significant processing overhead, and do not support the adaptive adjustment of routing node verification when the path policy changes.

In this paper, we design a new protocol for source and path validation, termed PSVM (Piece Split Validation Mechanism), which aims to confirm whether packets have been forwarded correctly in the data plane according to intended path policies. We engineer a flexible authentication structure called PIC, which translates the intended path policies (including the source) into a check set consisting of ordered path information pieces, instead of a whole value as usual. PSVM embeds PIC into the packet header as a validation standard before the session starts, and minimizes operations on PIC in nodes, which can significantly reduce packet processing latency and optimize communication cost. More importantly, by replacing PIC pieces, PSVM can support dynamic validation of actual packet transmission. Our contributions are three-fold:

- We introduce a basic PSVM protocol (in Section II-C). It does not update any PIC pieces in packet headers during packet verification, which enables a low overhead of processing in routers. This superiority is decisive because it only performs complex computation once, while the existing best-practice OPT still needs to do that twice.
- To the best of our knowledge, among existing source and path validation solutions, our dynamic PSVM (in Section III-A) is the first to support online migration and adjustment of node verification in the case of the intended path changes. It is trying to answer **(1)** whether it was possible to build one, **(2)** how to build one, and **(3)** what

it would cost to cope with the variable-path challenge. Indeed, it has good cost-effectiveness when intended path changes due to safety reasons.

- We implement our PSVM prototype on Click [11] and evaluate basic and dynamic PSVM protocols in different scenarios on a prototype-based real testbed (in Section V). The evaluation results indicate that the verification efficiency of PSVM in routing nodes is barely influenced by payload size or path length, and PSVM achieves a high throughput.

II. PSVM Basic Design

In this section, we first introduce PSVM architecture and give some assumptions, and then present the basic idea of PSVM validation and show its protocol design.

A. PSVM Architecture

PSVM is a validation protocol committed to providing capability to verify source authenticity and consistency of data plane forwarding and control plane path policies. To support the protocol, we employ PSVM architecture that separates general service functions from routing nodes, and its design is inspired by [3], [4], [12], some of which were actually used. Fig. 1 shows PSVM architecture, where a guarantee for the control plane path policies can be implemented by the provider's trusted agent, which is called Credible Guarantee Agent (CGA).

In PSVM architecture, CGAs interact with different nodes through PSVM protocol packets, e.g., responding to a node's request or error report. Each CGA manages some nodes. The master key of a node (Key_{Ni}) is only shared between the node (Ni) and its corresponding CGA. Then, CGA is able to derive a session symmetric key ($Key_{Ni}^{Session}$) from the node's master key (without seeking help from node Ni) [13] to calculate the authentication structure for source and path validation, which is treated as a path policy certificate. In particular, different providers may build one or more CGAs according to their needs, and CGA's service capability may be a virtual function module backed up or migrated on multiple CGAs, which enables the architecture with better availability and robustness.

B. Assumptions

We aim to achieve real-time session packet verification for data plane forwarding. We assume that existing secure routing protocols can enable end-nodes to learn the AS- or router-level intended path that packets will traverse. There are many ways that end-nodes (i.e.,the sender and receiver) could know path information, such as network topology analysis [14], obtained from some BGP related protocols [15], [16], or employing the existing control plane routing protocols [13] (especially Pathlet [17] or SCION [18] where the path is assigned by end-nodes).

Moreover, we assume that each node would use the self-certifying system (e.g., IDs as public keys) to sign the generated PSVM protocol packets containing requests, responses, and error reports according to security requirements. This

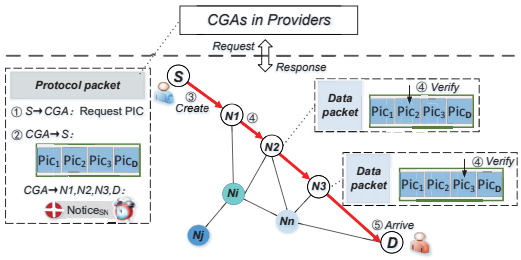


Fig. 1: The PSVM Architecture and Basic Workflow Overview.

paper temporarily does not discuss the loss of protocol packets, which may have a higher priority or be dispatched for multi-times and in multi-ways in practice to ensure a greater probability of being received. Furthermore, we suppose that node Ni and its CGA would use secret methods (such as Diffie-Hellman [19]) to share the master key (Key_{Ni}), which may be replaced if necessary to prevent cryptanalysis attacks. The session symmetric key of Ni ($Key_{Ni}^{Session}$) can be derived through pseudo-random operation functions [13] by Ni and its CGA, respectively, without long-term storage or re-exchange. Finally, we assume all the nodes in a session are loosely time synchronized (e.g., using NTP).

C. Basic PSVM Protocol

Compared to general source and path authentication protocols dealing with intended path information as a whole and lacking flexibility, we develop a piece by piece authentication structure called PIC, which divides the intended path information representing forwarding policies into a number of sequential pieces and each piece named as Pic logically expresses an directed adjacent relationship of two nodes.

Session initialization. (1) **Requesting PIC structure.** The sender needs to apply for a PIC structure of the session by submitting essential materials (including the intended path information and session time limit $\mathcal{T}_L^D$, etc.) to the relevant CGAs. After confirming the legitimacy of application, CGAs generate the Pics of PIC separately with the routing nodes' session keys and reply to the sender. Since all Pics are returned with a higher transmission priority, if the Pic belonging to a CGA has not arrived on time after a number of requests, the sender could request a CGA backup service for the piece to complete the corresponding Pic application.

(2) **Session notification.** CGAs will push tailored session notices $Notice_{Ni}$ (computed by Formula (1)) to each node on the path, which reminds the intended node of the session for subsequent verification.

$$Notice_{Ni} = Enc_{Key_{Ni}^{SN}}(SessionNum \parallel S \parallel D \parallel \mathcal{T}_L^D \parallel i), Sign_{SK_{CGA}}(Notice_{Ni}) \quad (1)$$

(3) **PSVM packet creation.** The sender configures session packets with a PSVM header (as shown in Fig. 2), and delivers them to the next hop.

- $\mathcal{SN}$: Hash of source S, destination D, session time limit in D ($\mathcal{T}_L^D$), and session symmetric key of S (Key_S^{SN}), denoted as $H(S \parallel D \parallel \mathcal{T}_L^D \parallel Key_S^{SN})$;

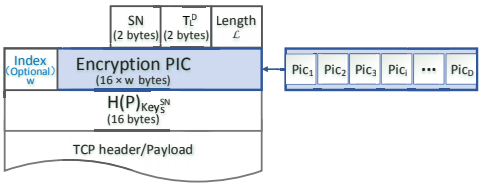


Fig. 2: PSVM Header. The value in parenthesis is the size of the field.

- INDEX (OPTIONAL FIELD): It is the directory of authentication structure PIC in the next field $Encryption\ PIC$, which gives $\mathcal{W}$ different labels to help different nodes find their corresponding Pics.
- ENCRYPTION PIC: This field contains $\mathcal{W}$ Pics that are generated by CGAs (computed as Formula (2)).

$$\begin{aligned} Pic_i &= Enc_{Key_{N_i}^{SN}}(N_{i-1}, N_{i+1}) \\ Pic_D &= Enc_{Key_D^{SN}}(D-1, S, Key_S^{SN}) \end{aligned} \tag{2}$$

- $H(P)_{Key_S^{SN}}$: This is the hash of the packet's payload using the session key Key_S^{SN} of S.

Session validation. (1) **Intermediate node validation.** When receiving a packet, an intermediate node N_i first checks its notice list to prevent unauthorized packets from being passed. For efficient verification, N_i only verifies its own Pic_i in the packet header, and the validation process (shown as Algorithm 1) does not need to update Pic_i.

Algorithm 1 Intermediate Node Validation Pseudocode

Require: N_i maintains a list of session $Notice_{Ni}$ in real time, which adds the received ones and removes the expired ones.

function VALIDATION BY INTERMEDIATE NODE N_i
  $\mathcal{SN}$' ← The $SessionNum$ in packet header
  $\mathcal{T}_L^D$' ← The $\mathcal{T}_L^D$ in packet header
  Fast look up $\mathcal{SN}$' in the list by $\mathcal{T}_L^D$' as a pointer
  if look up $\mathcal{SN}$' unsuccessfully **then**
    Return error report
  N'_{i-1} ← Collect the actual entry
  i ← Locate $N'_i s\ Pic_i$
  Compute Pic_i =$Dec_{Key_{N_i}^{SN}}(N_{i-1}, N_{i+1})$
  if (decode Pic_i successfully) and
    $(N'_{i-1} = N_{i-1})$ **then**
    Forward the packet
  else
    Drop the packet
    Return error report
  end if
end function

(2) **Destination validation.** If the packet has not been dropped halfway until reaching D and the check of $D-1$' has succeeded, D is assured that the packet's actual path is already bounded to the intended one in the light of property of Pics. Then, the conformance test of S' in packet header and S in Pic_D is a helper to confirm the claimed S' matches the expected S (shown as Algorithm 2). Finally, because the value in $H(P)_{Key_S^{SN}}$ field as a mark is calculated by S using the secret Key_S^{SN}, the correct result of re-calculating the actual hash of the packet's payload by Key_S^{SN} in Pic_D proves the packet payload is integrated, and re-tells that the packet is indeed originated from the expected S.

Algorithm 2 Destination Validation Pseudocode

function VALIDATION BY DESTINATION D
  $\mathcal{SN}$' ← The $SessionNum$ in packet header
  $\mathcal{T}_L^D$' ← The $\mathcal{T}_L^D$ in packet header
  Fast look up $\mathcal{SN}$' in the list by $\mathcal{T}_L^D$' as a pointer
  if look up $\mathcal{SN}$' unsuccessfully **then**
    Drop the packet
  D-1' ← Collect the actual entry
  S' ← The source in packet header
  $H(P)_{Key_S^{SN}}$ ← The field value in packet header
  Compute Pic_D =$Dec_{Key_D^{SN}}(D-1, S, Key_S^{SN})$
  if (decode Pic_D unsuccessfully) or
    $(D-1' \neq D-1)$ **then**
    Drop the packet
  Compute $H(P)'_{Key_S^{SN}}$ =$MAC_{Key_S^{SN}}(payload)$
  if $(S' \neq S)$ or
    $(H(P)'_{Key_S^{SN}} \neq H(P)_{Key_S^{SN}})$ **then**
    Drop the packet
  else
    Accept the packet
  end if
end function

III. REFINEMENT TO BASIC PSVM

In this section, we further introduce an improvement to basic PSVM called dynamic PSVM, and more detailed design of the improvement can be found in [20].

A. Dynamic PSVM

In order to execute real-time verification, existing source and path validation protocols embed the known test standard representing path policies in the packet in advance. Unfortunately, existing protocols do not take the factor of path policy changes into account, let alone that their test standard weave the path policy as a whole into a complex structure (e.g., an encrypted nested structure), which has a large cost of temporary modification. We further propose a dynamic validation method based on our flexible PIC structure to help basic PSVM overcome the above difficulties. As our PSVM is concerned with forwarding verification and is orthogonal to the problem of routing policy selection or even network failure recovery, dynamic PSVM is performed after the network environment is rebuilt. And we mostly assume that the sender is now ready for a candidate intended path.

Asking for new PIC. The sender S will request new Pics of PIC structure from CGAs in the same way as basic PSVM (Stage ① in Fig. 3), but CGAs will reply both S and the first cross node of the old and new paths, which is N_2 in stage ② or may be before N_2 in practice. The location of cross-node may have a slight impact on cost-effectiveness of dynamic PSVM. Together, CGAs will notify useless nodes on the old path to cancel the session and inform all nodes on the new intended path to add or update the session through $Notice_{Ni}$.

Dynamic verification. S launches the rest of session packets carrying new Pics in the PSVM header to the next hop. In addition to the check of previous hop information, N_2 also

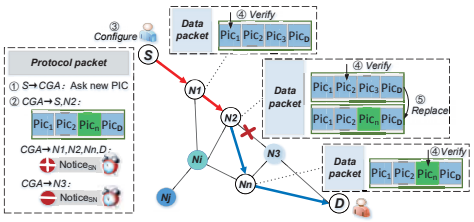


Fig. 3: The dynamic PSVM workflow overview.

needs to detect whether the intended next hop is N_3 or N_n in Pic_2 of currently arriving session packets. If it is N_3, representing that the packet takes old Pics and has been away from S before the dynamic change, N_2 will replace the old PSVM header into the new one and continue to send it to the normal downstream. Once there is N_n, N_2 will stop replacing, which means Pics in the packet header is already new and the adjustment process has been completed at the moment in stage ⑤. It is worth considering that in the case of network topology having no change but avoiding a malicious node or region, dynamic PSVM is more affordable and flexible.

IV. Security Analysis

This section discusses how our PSVM guards against a computation-limited attacker that may compromise the routing node at different stages of data transmission.

Malicious forwarding. If an attacker wants to redirect packets to any honest node N_j that is not on (or on) the intended path, it cannot create a legal Pic_j without the session key of N_j, and cannot pass the validation of N_j.

Modification of source address or payload. If a malicious node changes the source address, the destination D cannot verify Pic_D correctly which is encoded with the claimed source information and is frozen without the session key of D. Similarly, the destination D will be aware whether the payload is replaced when comparing the hash field $H(P)_{Key_S^{SN}}$.

Injection & Cloning attacks. Without the session keys, the malicious node can only inject packets of a valid session to the next hop on the session path, and can only inject packets with the same header and payload as the normal packets seen before. Then such injection or cloning could be mitigated by the session time limit $\mathcal{T}_L^D$ in PSVM.

Hidden threat. In PSVM, as long as the hidden node involves in the forwarding process, it will be found by the honest next hop using above approaches. It does not matter whether the packet contains the hidden node's mark.

Availability attack. When dealing with requests, CGA may be attacked by usability. A CGA victim can quickly migrate its capability to other reliable agents of providers. If there is a potential leak of session keys, the victim node is convenient to re-derive a session key, which just releases a replacing messages to the corresponding CGA instead of exchanging the session key itself.

Collusion. It is a conventional case that a malicious node cooperates with another node by sharing their session keys and may try to skip the honest nodes between them, or even redirect the packet to any other nodes outside the intended path. As long as the packet passes through an honest node during the delivery process, it cannot complete the PSVM legitimate validation of the honest node. Secondly, the skipped normal nodes will send the corresponding error report if no session packet is received within the session time limit.

In PSVM architecture, if a CGA is compromised to do bad things, such as revealing the session key, sending the notice or generating the Pics for the attacker. Based on the error reports, the destination can find: **1)** some nodes on the intended path do not receive session packets, or **2)** some nodes not on the intended session path are receiving session packets, which could help to discover potential troubles.

V. Implementation & Evaluation

We implement the prototype source and routing node of PSVM on a terminal computer and a modular software router Click [11], respectively. The source node system has one Intel(R) Core(TM) i5-4200u 1.60 GHz CPU, 4.0 GB memory and NIC 1000 Mbps. The routing node system has one Intel(R) Core(TM) i5-5200u 2.20 GHz CPU, 12.0 GB memory and employs a NIC with a maximum rate of 1000 Mbps. We conduct evaluations of our PSVM on a real testbed built upon the prototype using the normal IP routing performance of the Click router as the baseline, and compare our PSVM with the-state-of-the-art OPT [2].

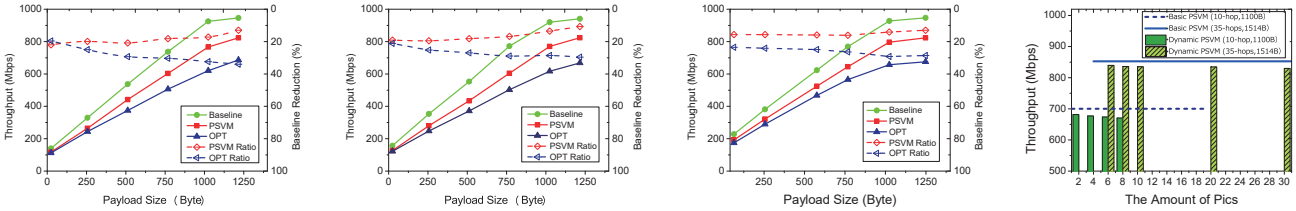
TABLE I: The Combinations of Experiment Parameters.

Num	Packet Sizes A \Payload Sizes B (Bytes)	Path Length (hops)	Inserted W_1 Pics \Modified W_2 Pics
1	$A \in \{1514\}$	$\{2, 4, 6, 8, 10, 15, 20, 25, 30, 35\}$	$W_1 \in \{2, 4, 6, 8, 10, 15, 20, 25, 30\}$
2	$B \in \{64, 256, 576, 768, 1024, 1248\}$	$\{4, 6, 10\}$	–
3	$A \in \{1100, 1514\}$	$\{10, 35\}$	$W_2 \in \{2, 4, 6, 8\}$ $W_2^* \in \{6, 8, 10, 20, 30\}$

TABLE II: The Processing Time and Throughput of the CGA and Source Operations. ℓ is the hops of path length.

Operations	Processing Time (μs) (a)	Throughput ($1/Processing\ Time$) (b)
CGN Pics Calculation	$0.88\ell + 0.08$	$1.12 * 10^6/(\ell + 0.09)$ (*Pic/s*)
Basic PSVM Packets Creation	$0.43 * 10^{-3}\ell + 0.51$	$2.29 * 10^9/(\ell + 1174.29)$ (*Pkt/s*)

Method and parameter setup. For fairness, we use the same 128-bit AES algorithm to compute the PSVM's authentication structure Pic and OPT's validation structure MAC, which can reach the throughput of 48 Gbps in the hardware implementation technology [21] and will not become a bottleneck of router forwarding. Moreover, we adopt the SHA-3 based 256-bit HMAC algorithm to generate hash strings of packet payload hash (which is truncated to a 128-bit value in packet header). Since either our PSVM or OPT needs to derive the session symmetric key, which is cached once calculated, we prefer to take the key from cache when doing this part. When the source sends packets to the routing node, we choose different path lengths, packet or payload sizes, the amount of modified



(a) 4-hop Path in Basic PSVM (b) 6-hop Path in Basic PSVM (c) 10-hop Path in Basic PSVM (d) Throughput vs. W_2

Fig. 4: The Average Throughput of Baseline, OPT, and all the Protocols of PSVM in the Routing Node. W_2 is the amount of modified Pics in the dynamic PSVM.

Pics and other parameters to estimate PSVM and OPT in time and space overhead (see Table I).

The source and CGA overhead. For cost measurement of CGA's generation of Pics and Pics insertion at source, we change the path length (according to combination 1 in Table I), count the total processing time repeatedly, and fit the functions of path length and processing time in Table II by using the least squares linear regression where $R^2 > 0.99$. As shown in Table II, the cost of Pics calculation would increase by about $0.88\ \mu s$ per hop and the generation rate of the PIC structure for a 15-hop path is approximately $74.2K$ Pics/s.

The routing node throughput in basic PSVM. We investigate the throughput performance (according to parameters of combination 2 in Table I), and compute the throughput reduction ratio of basic PSVM and OPT using the baseline throughput. Fig. 4 (a) to (c) reveal that: **1)** the throughput of basic PSVM outperforms that of OPT at different path lengths. The maximum throughput of basic PSVM is about 823 Mbit/s in 10-hop path, by comparison, the maximum throughput of OPT is about 675 Mbit/s. This result is mainly caused by performing twice complex calculation for each packet in OPT (more than PSVM). **2)** despite the path length (or payload size) changes, the throughput (or throughput reduction ratio) of basic PSVM and OPT are barely influenced because their verification operations (such as computing Pics or MACs) are irrelevant to the path length (or payload size) in a routing node. With reference to the baseline throughput, the average throughput reduction ratio for basic PSVM and OPT are 13% and 28%, respectively. **3)** in the destination, basic PSVM's operations are still independent of path lengths, but OPT's are proportional to path hops.

The routing node throughput in dynamic PSVM. Dynamic PSVM contains a cross node that is different from other nodes on the path and not only performs verification operations but also does new Pics replacement during path adjustment. To better understand the cost of cross node, we pick out two reference lines: **a) line ⓐ.** 656 Mbit/s throughput of basic PSVM node (a 10-hop path, 1100B packet size). **b) line ⓑ.** 839 Mbit/s throughput of basic PSVM node (a 35-hop path, 1514B packet size). Then, under the same conditions of reference line ⓐ and line ⓑ (i.e., with the same path length and packet size in the session), we examine the throughput of the cross node when it needs to replace W_2 and W_2^* Pics detailed in combination 3 of Table I, respectively. It is observed in Fig. 4(d) that for a 10-hop path with 8 Pics needed to be replaced, the throughput of the cross-node is 637 Mb/s and the reduction ratio is 2.8% compared to the reference line ⓐ. For a 35-hop path with 30 Pics to be replaced, the throughput of the cross-node is 819 Mb/s and the reduction ratio is 2.4% compared to the reference line ⓑ.

The routing node communication and storage overhead. For communication overhead, OPT and basic PSVM have an additional protocol header of $52 + 16 * \mathcal{L}$ and $21 + 16 * \mathcal{L}$ bytes (where $\mathcal{L}$ is the path length), respectively. OPT requires a larger number of bytes, so it would consume more bandwidths. In a session, the total number of bytes of Pics and notices generated by CGAs can be calculated as $16*\mathcal{L}+32*\mathcal{L} = 48*\mathcal{L}$. For IP path with the average length of 13.11 hops, it will be no more than 630B which is shared by the entire session.

For storage overhead, PSVM only store one master key for a long time to serve the derivation of session keys. Furthermore, we select a sample trace of CAIDA [22] to translate the cost of storing the notice list on a routing node in PSVM. The sample trace shows that the total number of application flows observed is 12.90463K on average at one time in a routing node. When a session notice is calculated in 32B, the total size of the notice list for all application flows stored at the same time will not exceed 0.42MB, which consumes little compared to today's SRAM capabilities [23].

VI. Discussion

Compatibility. As an extension of a common routing protocol, PSVM can act in conjunction with routing protocols (e.g., BGP, OSPF, or Pathlet). PSVM can accommodate IPv4 and IPv6 address format transmission. In IPv4, we recommend using a method of embedding Pics with random probability to simplify basic PSVM, where the PSVM header can be placed in the optional field of IPv4 headspace.

The SDN scenario. For PSVM deployment, the SDN network is a natural environment reflected in the following two aspects: **1)** The controller of SDN can bear the function of CGAs. **2)** We suggest that a PSVM header in SDN only keeps necessary information as the protocol identifier, and Index and Pics can be merged into the session notice. In addition to storing the original notice information, the routing node also needs to store its Pic. As shown in a sample trace of CAIDA [22] (in Section V), with 16 B Pic and 1 B Index, fitting the critical verification information of all application flows would need less than 0.22 MB. Yet, the PSVM overhead in an SDN packet is only 21 B, which is a very exciting result.

VII. Related Work

Path detection and Fault localization. Many researches provide path validation by making intermediate routers collect packet information [24], and infer path properties of interest [25]. Fault localization is recognized as a high-quality online service, since it enables receivers to efficiently localize faulty links [26]–[28]. In practice, most of these systems are expensive to deploy, and they all require a certain amount of extra storage overhead for intermediate routers.

Source and path validation. Current techniques for both source authentication and path verification have been proposed in ICING [3], the Origin and Path Trace (OPT) [2], Orthogonal Sequence Verification (OSV) [4], and PPV [29]. In ICING, it requires each router to verify the optimized cryptographic construction PoP for all upstream nodes and generate new PoPs for every downstream node with the secret keys shared among all intermediate routers, causing considerable communication and computation overhead. OPT improves the processing performance of intermediate nodes. OSV uses orthogonal sequences (instead of cryptographic structure) as credentials that identify the source and verify the path. PPV attempts to reduce the communication overhead by means of probabilistic packet marking, but it cannot carry out real-time verification at intermediate nodes. Additionally, the above mechanisms do not support adaptive adjustments in the session upon environment requirement.

VIII. Conclusion

In this paper, PSVM tackles an important missing piece in current networks — performing a flexible verification for source authenticity and path consistency based on intended polices. It achieves many advantages and is superior to related protocols in terms of throughput performance. We anticipate our PSVM will promote the work of source and path validation, stepping forward to the field of practical applications.

Acknowledgments

This work was in part supported by the National Key R&D Program of China with No. 2018YFB0803405, National Science Foundation for Distinguished Young Scholars of China with No. 61825204, National Natural Science Foundation of China with No. 61932016, No. 61972039, No. 61572278, Beijing Natural Science Foundation with No. 4192050, Beijing Outstanding Young Scientist Program with No. BJJWZYJH01201910003011, BNRist with No. BNR2019RC01011, PCL Future Greater-Bay Area Network Facilities for Largescale Experiments and Applications with No. LZC0019.

References

[1] Y. Zhao, H. Wang, H. Su, L. Zhang, R. Zhang, D. Wang, and K. Xu, "Understand love of variety in wireless data market under sponsored data plans," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 38, no. 4, pp. 766–781, 2020.

[2] T. H.-J. Kim, C. Basescu, L. Jia, S. B. Lee, Y.-C. Hu, and A. Perrig, "Lightweight source authentication and path validation," in *Proceedings of ACM SIGCOMM*, 2014, pp. 271–282.

[3] J. Naous, M. Walfish, A. Nicolosi, D. Mazières, M. Miller, and A. Seehra, "Verifying and enforcing network paths with icing," in *Proceedings of ACM CoNEXT*, 2011, p. 30.

[4] H. Cai and T. Wolf, "Source authentication and path validation with orthogonal network capabilities," in *Proceedings of IEEE INFOCOM WKSHPS*, 2015, pp. 111–112.

[5] D. Madory, "Uk traffic diverted through ukraine," http://research.dyn.com/2015/03/uk-traffic-diverted-ukraine.

[6] M. Shen, B. Ma, L. Zhu, R. Mijumbi, X. Du, and J. Hu, "Cloud-based approximate constrained shortest distance queries over encrypted graphs with privacy protection," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 13, no. 4, pp. 940–953, 2018.

[7] J. Wu, G. Ren, and X. Li, "Source address validation: Architecture and protocol design," in *Proceedings of IEEE ICNP*, 2007, pp. 276–283.

[8] M. Shen, H. Liu, L. Zhu, K. Xu, H. Yu, X. Du, and M. Guizani, "Blockchain-assisted secure device authentication for cross-domain industrial iot," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 38, no. 5, pp. 942–954, 2020.

[9] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Practical network support for ip traceback," in *Proceedings of ACM SIGCOMM*, 2000, pp. 295–306.

[10] A. Yaar, A. Perrig, and D. Song, "Pi: A path identification mechanism to defend against ddos attacks," in *Proceedings of IEEE SP*, 2003, pp. 93–107.

[11] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek, "The click modular router," *ACM Transactions on Computer Systems (TOCS)*, vol. 18, no. 3, pp. 263–297, 2000.

[12] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Proceedings of Crypto*, 2001, pp. 213–229.

[13] B. Raghavan and A. C. Snoeren, "A system for authenticated policy-compliant routing," in *Proceedings of ACM SIGCOMM*, 2004, pp. 167–178.

[14] R. Nithyanand, O. Starov, A. Zair, P. Gill, and M. Schapira, "Measuring and mitigating as-level adversaries against tor," 01 2016.

[15] S. Kent, C. Lynn, and K. Seo, "Secure border gateway protocol," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 18, no. 4, pp. 582–592, 2002.

[16] Y. C. Hu, A. Perrig, and M. A. Sirbu, "Spv: secure path vector routing for securing bgp," in *Proceedings of ACM SIGCOMM*, 2004, pp. 179–192.

[17] P. Godfrey, I. Ganichev, S. Shenker, and I. Stoica, "Pathlet routing," pp. 111–122, 2009.

[18] X. Zhang, H.-C. Hsiao, G. Hasker, H. Chan, A. Perrig, and D. G. Andersen, "Scion: Scalability, control, and isolation on next-generation networks," in *Proceedings of IEEE SP*, 2011, pp. 212–227.

[19] V. Boyko, P. MacKenzie, and S. Patel, "Provably secure password-authenticated key exchange using diffie-hellman," in *Proceedings of Eurocrypt*, 2000, pp. 156–171.

[20] "Technical Report," https://www.dropbox.com/s/c7fbtn0x6dx6nwc/TR_PSVM.pdf?dl=0.

[21] "Aes fast," http://cavium.com/processor_security_nitroxII.htm.

[22] "Passive monitor: equinix-chicago," http://www.caida.org/data/monitors/passive-equinix-chicago.xml.

[23] "Reliable, high-performance synchronous srams from cypress," http://www.cypress.com/products/synchronous-sram.

[24] S. Narayana, M. Tahmasbi, J. Rexford, and D. Walker, "Compiling path queries," in *Proceedings of NSDI*, 2016, pp. 207–222.

[25] D. Levin, Y. Lee, L. Valenta, Z. Li, V. Lai, C. Lumezanu, N. Spring, and B. Bhattacharjee, "Alibi routing," in *Proceedings of ACM SIGCOMM*, 2015, pp. 611–624.

[26] X. Zhang, Z. Zhou, H. C. Hsiao, T. Kim, P. Tague, and A. Perrig, "Shortmac: Efficient data-plane fault localization," 2011.

[27] C. Basescu, Y. H. Lin, H. Zhang, and A. Perrig, "High-speed inter-domain fault localization," in *Proceedings of IEEE SP*, 2016.

[28] B. Wu, K. Xu, Q. Li, B. Liu, S. Ren, F. Yang, M. Shen, and K. Ren, "Rfl: Robust fault localization on unreliable communication channels," *Computer Networks*, 2019.

[29] B. Wu, K. Xu, Q. Li, Z. Liu, Y. Hu, M. J. Reed, M. Shen, and F. Yang, "Enabling efficient source and path verification via probabilistic packet marking," in *Proceedings of IEEE/ACM IWQoS*, 2018.