UniSAV: A Unified Framework for Internet-Scale Source Address Validation

Lancheng Qin
Tsinghua University

Libin Liu
Zhongguancun Laboratory

Li Chen
Zhongguancun Laboratory

Dan Li
Tsinghua University

Yuqian Shi
Zhongguancun Laboratory

Hongbing Yang
Zhongguancun Laboratory

ABSTRACT

To mitigate the threats of source address spoofing, many source address validation (SAV) solutions have been proposed over the past few years. However, none of them are widely deployed in practice due to lack of understanding, lack of open source implementation, and performance concerns. To address these problems, we develop a unified framework UniSAV to facilitate the understanding, design, implementation, and evaluation of existing and future SAV solutions. With UniSAV, we implement existing SAV solutions and evaluate their performance in real topologies. UniSAV further helps us to design and implement a new SAV solution to improve the performance upon existing SAV solutions.

CCS CONCEPTS

• **Security and privacy** → **Network security**; • **Software and its engineering** → **Software organization and properties**;

ACM Reference Format:
Lancheng Qin, Libin Liu, Li Chen, Dan Li, Yuqian Shi, and Hongbing Yang. 2024. UniSAV: A Unified Framework for Internet-Scale Source Address Validation. In *Applied Networking Research Workshop (ANRW 24), July 23, 2024, Vancouver, AA, Canada.* ACM, Vancouver, Canada, 7 pages. https://doi.org/10.1145/3673422.3674888

1 INTRODUCTION

Source address spoofing is one of the most serious security threats in today's Internet. By sending packets with spoofing source IP addresses, attackers can carry out various malicious attacks while hiding their real identities [4, 29]. In particular, it is the main attack vector for large-scale Distributed Denial of Service (DDoS) attacks [6, 8–10].

To prevent source address spoofing, the Internet Engineering Task Force (IETF) published the Best Current Practice (BCP) for SAV (*i.e.*, BCP38 [18] and BCP84 [15, 36]) more than 20 years ago. The Mutually Agreed Norms for Routing Security (MANRS) [3], a global initiative supported by the Internet Society (ISOC), has been calling on network operators to deploy SAV and other routing security mechanisms since 2014 [7, 32, 33, 38]. In June 2022, the SAVNET working group [2] was formed in IETF to guide the development of new SAV mechanisms, including distributed protocols such as DSAV [25].

However, previous SAV measurement studies [16, 17, 21, 30] show that the adoption of SAV is worryingly low, especially the adoption of inbound SAV. A recent study [28] reveals that many network operators do not adopt SAV because they lack knowledge of existing SAV mechanisms or are concerned about possible performance issues after adopting SAV. Since there is no widely accepted and trusted methodology to evaluate the performance of SAV mechanisms, network operators are unable to determine the return on investment and effectiveness of different SAV mechanisms, deterring them from adopting SAV.

To promote the adoption of SAV, we develop a unified framework UniSAV to facilitate the design, implementation, and evaluation of existing and future SAV mechanisms. This framework features a unified abstraction of existing SAV mechanisms and offers a streamlined approach to the implementation and development of SAV mechanisms. UniSAV is open-source[1], and we have implemented almost all existing SAV mechanisms with it. It has a container-based runtime that can run different SAV scenarios, which greatly helps network operators understand and evaluate the performance of different SAV mechanisms. For comprehensive testing, we collaborate with an ISP and a device vendor to create SAVBench, a testing benchmark for SAV mechanisms, which includes 300 real topologies in three scenario categories. UniSAV enables us to reveal previously unknown issues of

ANRW 24, July 23, 2024, Vancouver, AA, Canada
© 2024 Association for Computing Machinery.
ACM ISBN 979-8-4007-0723-0/24/07.
https://doi.org/10.1145/3673422.3674888

[1]https://github.com/SAV-Open-Playground

Existing SAV Mechanisms	SAV Information Source			SAV Rule Format	
	Manual Configuration	Third-party Public Database	Communication between ASes	(Prefix, Incoming Interface)	(Prefix, Crypt. Key)
uRPF [15, 36], SAVE [26], DSAV [25]			✓	✓	
Passport [27], EPIC [23], PISKES [34]			✓		✓
BAR-SAV [35]		✓	✓	✓	
Ingress Filtering [18]	✓			✓	

Table 1: Summary of existing SAV mechanisms.

existing SAV mechanisms and helps us to design and test new SAV mechanisms.

In this paper, we make the following contributions.

- We design and develop UniSAV, the open source framework for SAV, and implement nine SAV mechanisms. This framework provides useful common utilities for all stages of any SAV mechanism, greatly reducing the effort of designing a new one.
- We develop a containerized runtime environment for UniSAV that can run large-scale scenarios of 200 ASes on a single server. We develop the first SAV testing benchmark, SAVBench, for ISPs to validate SAV mechanisms.
- We design E-DSAV, a new SAV mechanism that enhances DSAV. With the UniSAV's runtime environment, we compare E-DSAV with DSAV and conclude that E-DSAV has 52.99% higher control plane throughput while achieving the same accuracy.

2 BACKGROUND AND MOTIVATION

2.1 Existing SAV Mechanisms

Existing SAV mechanisms encompass the same basic functions for collecting SAV information, generating SAV rules, and executing SAV filtering. However, they can be different in SAV information source or SAV rule format. Table 1 outlines existing SAV mechanisms.

uRPF-based SAV mechanisms [15, 36] and **distributed SAV protocols** (such as SAVE [26] and DSAV [25]) determine valid incoming interfaces for source prefixes by using information communicated between ASes. Specifically, uRPF-based SAV mechanisms use routing information learned from the routing protocol, while distributed SAV protocols allow each AS to notify the incoming direction of its source prefixes to other ASes by sending SAV messages.

Cryptographic SAV mechanisms (such as Passport [27], EPIC [23], and PISKES [34]) require ASes to communicate their cryptographic keys between each other through the key exchange protocol. When sending a packet, the sending AS calculates a tag by using the cryptographic key and add the tag in the packet. The receiving AS checks the correctness of the tag in the packet to perform SAV.

BAR-SAV [35] is a refined version of EFP-uRPF [36]. In addition to using inter-domain routing information communicated among ASes, BAR-SAV uses information from third-party public databases, such as the RPKI ROA repositories or the ASPA repositories.

Ingress Filtering [18] relies on ACL rules to filter packets based on their source addresses. The ACL rules are manually configured and updated by network operators.

2.2 Problems of SAV Deployment

A discussion organized by ISOC in 2015 concluded that the significant diversity in SAV implementation had greatly hindered the deployment of SAV, and "Implementation of anti-spoofing mechanisms and mechanisms in key open source projects is equally important" [5]. A more recent study [28] indicates that network operators do not deploy SAV mainly because they lack knowledge of existing SAV mechanisms or are concerned about possible performance issues after adopting SAV. The Internet community is suggested to put more effort into providing better SAV supporting resources like software and technical documents [28].

Based on the results of previous work, we can summarize three major problems of SAV deployment:

P1 **Lack of understanding.** Many network operators lack the technical knowledge, understanding, and practical experience. They do not know how existing SAV mechanisms work, which SAV mechanism is the most suitable for their networks, or how to deploy or operate a specific SAV mechanism.

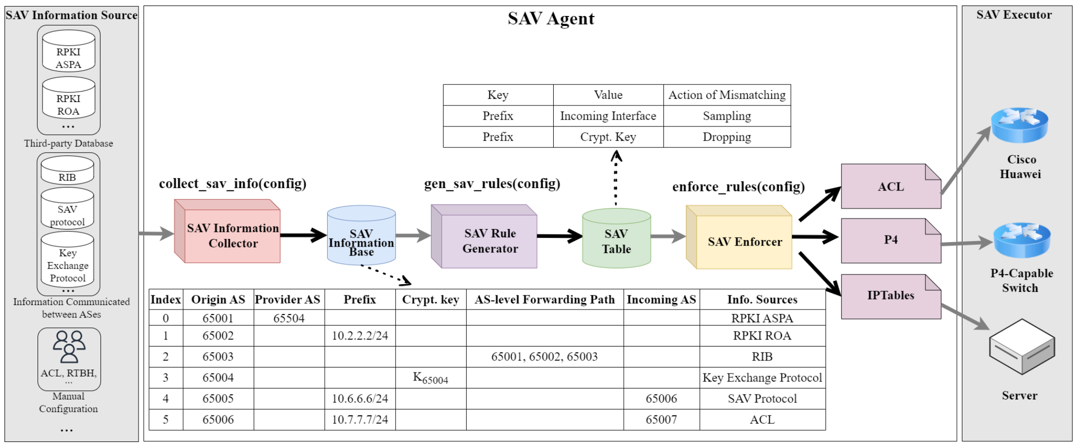
P2 **Lack of open source implementation.** Since there is limited open source effort on SAV implementation, it is difficult to form an acknowledged baseline standard, leading to differences in understanding and implementation of the same SAV mechanism.

P3 **Performance concerns.** Due to the lack of a publicly available testbed, people cannot test and evaluate the performance of different SAV mechanisms. Without sufficient tests, network operators hesitate to deploy SAV mechanisms in their networks.

3 DESIGN OF UNISAV

3.1 Design Overview of UniSAV

To address P1, we revisit existing SAV mechanisms from a high-level perspective and summarize a unified architecture that can cover all existing SAV mechanisms and possible future ones. Network operators, equipment vendors, and researchers can better understand the basic working principles of different SAV mechanisms through the unified SAV architecture.



Key	Value	Action of Mismatching
Prefix	Incoming Interface	Sampling
Prefix	Crypt. Key	Dropping

Index	Origin AS	Provider AS	Prefix	Crypt. key	AS-level Forwarding Path	Incoming AS	Info. Sources
0	65001	65504					RPKI ASPA
1	65002		10.2.2.2/24				RPKI ROA
2	65003				65001, 65002, 65003		RIB
3	65004			K_{65004}			Key Exchange Protocol
4	65005		10.6.6.6/24			65006	SAV Protocol
5	65006		10.7.7.7/24			65007	ACL

Figure 1: UniSAV architecture.

To address P2, we design and implement an open source platform to implement existing SAV mechanisms and develop new SAV mechanisms. Through this platform, anyone working in the area of SAV can participate in the development and evaluation of SAV mechanisms.

To address P3, we develop a containerized runtime environment to evaluate the control-plane and data-plane overhead of different SAV mechanisms. Based on the evaluation results, network operators can choose to deploy the SAV mechanism that best suits their hardware conditions.

3.2 UniSAV Architecture

As shown in Figure 1, the core concept of UniSAV is the SAV Agent, which encapsulates a SAV mechanism. It has three modules (*i.e.*, SAV information collector, SAV rule generator, and SAV enforcer) and two essential data structures (*i.e.*, SAV information base (SIB) and SAV table).

SAV Information Collector: This module is used to collect SAV information from one or more SAV information sources, such as manual configuration, FIB, RIB, RPKI, key exchange protocol, or SAV protocol. After that, this module will consolidate the collected information in the SIB. Information from different SAV information sources may vary in format and content. Figure 1 shows an example of SIB which contains SAV information collected from 6 SAV information sources. The "Origin AS" in SIB is the AS that registers the ROA or ASPA object, originates the message through a protocol, or manually configures SAV rules.

SIB: SAV Information Base (SIB) is the data structure that stores SAV information consolidated by SAV Information Collector.

SAV Rule Generator: This module is used to process the SAV information stored in the SIB and output a SAV table. Even for the same SIB, the SAV rule generators of different SAV mechanisms will generate different SAV tables.

SAV Table: SAV table is the data structure that stores SAV rules generated by SAV Rule Generator. In existing SAV mechanisms, it either maps the source prefix to a cryptographic key or maps the source prefix to a legitimate incoming interface.

SAV Enforcer: This module is used to perform SAV on data plane by using the SAV table. It checks the source address of each IP packet against the SAV table, and finds the binding information (cryptographic key or legitimate incoming interface) for the source address. Then, it uses the binding information to validate the authenticity of the source address. For packets identified as source-spoofed, this module defines three different operations, i.e., sampling, rate limiting, and dropping.

3.3 UniSAV Implementation

Following the architecture, we implement a software platform for SAV implementation and evaluation.

Implementation of SAV Information Collector: The SAV information collector adopts the command line interface (CLI) to collect the information from different SAV information sources, and we develop a common interface to hide details of interacting with different software and hardware routers. For example, in an Ubuntu server, UniSAV utilizes the `route -n -F` command to obtain all the routes from FIB. The SAV information collector contains utilities that can communicate with *Krill* [11] to access the ROA objects and ASPA objects.

Implementation of SAV Ruler Generator: With the SAV-related information collected in SIB, the SAV rule generator then processes the SIB to produce SAV rules, which are then stored in the SAV Table. Users can develop their own rule generator by overriding the `gen_sav_rules(·)` function. Basically, the SAV table records the prefix and its corresponding legitimate incoming interface or its mapped cryptographic key, as well as the intended action to take.

Implementation of SAV Enforcer: The SAV enforcer convert the SAV rules in the generated SAV Table to the policies supported by the prevalent data plane executors, such as access control list (ACL), P4 [12], and IPTables [31]. Network operators can specify the types of SAV operations utilized on the data plane. The data plane executors can enable and run the SAV rules on multiple SAV executors without any modifications. UniSAV currently supports three types of executors: iptables-based, ACL-based (commercial router from Cisco, Juniper, Huawei, etc.), and P4-based.

4 IMPLEMENTING SAV MECHANISMS

With UniSAV, we implement eight existing SAV mechanisms and a new SAV mechanism called E-DSAV on the basis of DSAV. Since the procedures of uRPF-based mechanisms are relatively simple, we mainly introduce our implementations of BAR-SAV [35], Passport [27], DSAV [25], and E-DSAV in the following.

4.1 Implementation of BAR-SAV

BAR-SAV uses the SAV-related information from multiple SAV information sources, i.e., RPKI ASPA [14], RPKI ROA [24], and/or BGP UPDATE data. As described in §3.3, UniSAV can obtain the SAV-related information from FIB, RIB, and RPKI. Then, we store the SAV-related information in the unified SAV information base (SIB). Next, with the information in the SIB and BAR-SAV core algorithm, we generate the SAV rules and store them in the SAV table. Finally, we enforce the generated SAV rules in the data plane and perform SAV with ACL.

4.2 Implementation of Passport

Passport is one of the representative methods of cryptographic SAV. It needs to communicate the symmetric keys used for SAV between ASes with a control plane protocol. As a result, to implement Passport, we extend UniSAV's function for SAV information collection and allow it to communicate with other SAV Agents. In our implementation of Passport, each AS generates a Diffie-Hellman [20] public/private value pair, such as (b_i, r_i) for AS_i. Each AS proactively sends the public value b to all other ASes based on the BGP neighbor information. That is, each AS sends its own public value b or forwards it to its BGP neighbors.

4.3 Implementation of DSAV

DSAV requires each AS to notify the incoming direction of its source prefixes to other ASes by sending control plane messages. The message consists of two main fields: i) the source prefix field is used to notify the range of source prefixes of the origin AS; ii) the propagation scope field is used to guide the propagation of messages along the direction of data plane forwarding paths. ASes along the forwarding path will receive this message and identify legitimate incoming interfaces for the corresponding source prefixes.

Following the detailed design of DSAV, we implement DSAV with UniSAV by extending the BGP UPDATE message to propagate DSAV messages. Our evaluations in §5.2 show that, compared to other existing mechanisms, DSAV achieves higher accuracy in challenging routing scenarios. However, UniSAV allows us to find that DSAV has limitations in overhead and performance. Given the large number of prefixes in inter-domain networks, it is a significant challenge for DSAV to transmit so many inter-domain prefixes between ASes, which will take up a lot of bandwidth resources. We reveal this using experiments in §5.3. In addition, propagating the SAV messages through BGP can affect the performance of BGP processing, which is evaluated in §5.5.

4.4 Implementation of E-DSAV

To address the limitations of DSAV, we design and implement a new SAV mechanism E-DSAV with UniSAV. E-DSAV makes the following two main improvements on DSAV:

Message Content Modification: To reduce the message size, E-DSAV replaces source prefix field with an ASN (AS Number) of the origin AS. The SAV agent can obtain the corresponding source prefixes of an ASN by checking local FIB or querying RPKI ROA repositories. Considering a large AS can have hundreds of prefixes, this modification can significantly reduce the message size. For the propagation scope field, since inter-domain RIB has already recorded the ASN path (*i.e.*, the AS-level forwarding path shown in Figure 1) to each destination prefix, the propagation scope field can directly carry the ASN path of each best route.

Decoupling Control and Data Channels: Transmitting SAV information through BGP can affect the performance of the control plane of the underlying router. Thus, we decouple the E-DSAV control channel from the data channel. Only the control channel reuses the BGP connection of the underlying router. E-DSAV uses the control channel to setup data channels with neighbors, and then uses the data channel to transmit SAV information (*i.e.*, information in source prefix field and propagation scope field). We choose to use QUIC [22] as the transport protocol for the data channel, and extend the UniSAV Information Collector to implement both channels.

5 EVALUATION

We implement a prototype of UniSAV with BIRD [13] as the underlying software router,and then evaluate existing SAV mechanisms and E-DSAV in various aspects as well as the scalability of UniSAV testbed.

5.1 Experimental Setup

Testbed. We use an x86 server machine to deploy the testbed experiments, which has two 2.2GHz 26-core Intel Xeon Gold 5320 CPUs, one 256GB DDR4 RAM, two 1TB SSDs, and one

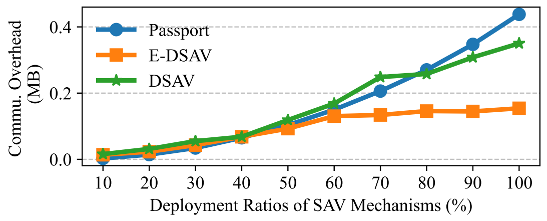


Figure 2: The total message size for SAV information communication under different deployment ratios.

12TB SAS HDDs. The server machine serves as the infrastructure to run the container-based network, where each container uses Docker 24.0.2 with the image ubuntu:22.04 and is used to emulate an AS. Within each container, we run BIRD 2.0.12 as the software router to perform the functions of AS border routers, and use iptables 1.8.7 as the data plane executor to filter packets.

SAVBench. We design the first testing benchmark for SAV, SAVBench. To build this benchmark, we first get the Internet topology from CAIDA [1]. Then we randomly sample from the full topology and obtain sub-graphs with different network sizes. Each sub-graph is a connected component of the full topology. We assign initial routing policies based on the business relationship and the valley-free principle [19]. For each sub-graph, we create three test-cases, one for a different scenario: symmetric routing, NO_EXPORT, and direct server return (DSR). The three test-cases are created according to the description in the problem statement document of SAVNET working group [37]. The symmetric routing test-case does not add any additional policies. For NO_EXPORT scenarios, we choose an AS with multiple providers, and let it add NO_EXPORT community in its BGP updates announced to one of its providers. For DSR scenarios, we randomly choose a prefix announced by an AS and let another AS which does not announce this prefix send data packets using source IP addresses in this prefix. SAVBench contains 300 test-cases of topologies with various sizes. In the following, we use the test-cases with 50 ASes to test different SAV mechanisms, except for the scalability experiments in §5.6.

5.2 SAV Accuracy

In order to comprehensively measure the accuracy of different SAV mechanisms, we test each SAV mechanism in all test-cases of SAVBench. In each test-case, only the victim AS and the validation AS (an upstream AS which provides security protection services for the victim AS) deploy SAV. The attacker AS sends spoofing traffic that uses source addresses belonging to the victim AS to the validation AS. For each SAV mechanism implemented on UniSAV, we respectively test whether it improperly blocks legitimate traffic originated, and whether it improperly admits the spoofing traffic originated from the attacker AS.

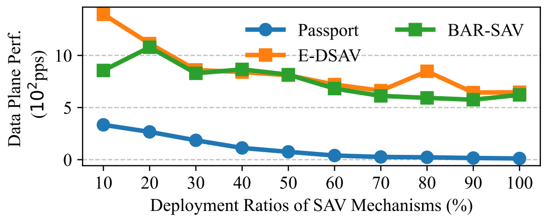


Figure 3: The data plane forwarding performance of the SAV mechanisms under different deployment ratios.

Table 2 shows the results of different SAV mechanisms in different scenarios. The results are exactly as expected from our theoretical analysis. Strict uRPF, FP-uRPF, EFP-uRPF, and BAR-SAV may improperly block legitimate traffic or improper admit spoofing traffic in the two challenging scenarios, while Passport, DSAV, and E-DSAV do not appear improper block or improper admit problems in all three scenarios.

5.3 Communication Overhead

Passport, DSAV, and E-DSAV require ASes to communicate SAV information between each other. To evaluate the communication overhead of each mechanism, we quantify the total size of the messages for SAV information communication during the convergence period.

Figure 2 shows the communication overhead of each mechanism under different deployment ratios. The total size of messages increases proportionally with the increase in deployment ratio. This trend aligns with the increased need for SAV information exchange, encompassing elements such as cryptographic keys, prefixes, or ASN paths among the deployed ASes. As shown in Figure 2, Passport has larger communication overhead than DSAV and E-DSAV when the deployment ratio is high. In addition, E-DSAV exhibits a much smaller message size compared to DSAV, a distinction attributed to the message content modification.

5.4 Data Plane Performance

We evaluate the data plane forwarding performance using SAV rules generated by various SAV mechanisms. Here, we employ iptables as a mechanism to execute SAV within the data plane. Specifically, we implement a traffic generation tool to generate packets with fixed 1.5KB size to evaluate the data plane forwarding performance in terms of packets per second.

Figure 3 shows the results of BAR-SAV, Passport, and E-DSAV. The results indicate that Passport always has a more significant impact on data plane forwarding performance than other SAV mechanisms. It is because cryptographical SAV requires the router to perform cryptographic computation on each packet, which greatly increases the processing overhead in data plane. For other SAV mechanisms, they just check the source address and actual incoming interface of

Scenarios	Strict uRPF	FP-uRPF	Loose uRPF	EFP-uRPF A	EFP-uRPF B	BAR-SAV	Passport	DSAV	E-DSAV
Symmetric routing	None	None	IA	None	IA	None	None	None	None
NO-EXPORT	IB	IB	IA	IB	IA	IB	None	None	None
Direct Server Return	IB	IB	IA	IB	IA & IB	IB	None	None	None

Table 2: The accuracy problems of different SAV mechanisms in three kinds of routing scenarios (None: test-case exhibits No Problem; IB: test-case exhibits Improper Block Problem; IA: test-case exhibits Improper Admit Problem).

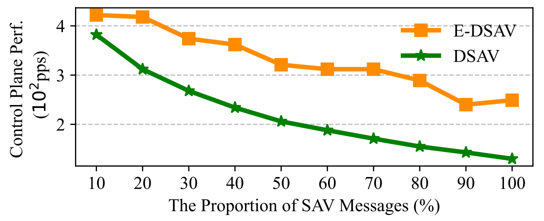


Figure 4: The control plane performance for processing pure BGP messages with varying proportions of SAV message.

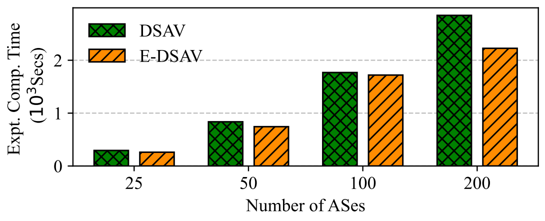


Figure 5: The experiment completion time of UniSAV under different network scales.

each packet against the SAV table, which is a lightweight table query operation. Besides, the data plane forwarding performance of each SAV mechanism decreases as the deployment ratio increases. This is because the size of the SAV table within each AS increases with the increase of deployment ratio, larger SAV table results in longer query time for each incoming packet.

5.5 Influence on BGP Processing

Considering DSAV and E-DSAV may occupy resources for processing BGP messages, we evaluate their impact on the performance of processing pure BGP messages. We first fix the number of messages that need to be transmitted between ASes. Some messages only carry BGP data (*i.e.*, pure BGP messages) while other messages carry SAV information (*i.e.*, SAV messages). By varying the number of messages that carry SAV information, we can vary the proportion of SAV messages. Then, we measure the control plane performance for processing pure BGP messages under different proportions of SAV messages.

Figure 4 shows the control plane performance for processing pure BGP messages in terms of packets per second. Both DSAV and E-DSAV impact the efficiency of the control plane in dealing with pure BGP messages. However, the underlying reasons for these effects differ between the two approaches. For E-DSAV, the limitations arise from resource constraints within each container. The transmission of SAV information places additional demands on these resources, thus reducing the capacity available to process pure BGP messages. For DSAV, it not only introduces complexities in the processing logic for pure BGP messages but also necessitates additional resources for parsing the delivered SAV messages. Instead, E-DSAV uses the data channel to deliver SAV messages which does not affect the control channel of BGP. Consequently, DSAV exhibits a more pronounced negative influence on the control plane performance of BGP processing. On average, E-DSAV improves the control plane performance of BGP processing by 52.99% compared to DSAV.

5.6 Testbed Scalability

We evaluate the overall scalability of UniSAV testbed. We fix the maximum experiment memory as 256GB and vary the network scales by increasing the number of ASes for the testbed experiments, and then calculate the experiment completion time. The experiment completion time is the longest time elapsed from launching the Docker environment to generating complete SAV Table among all ASes, which includes the startup time of Docker environment, the building time of network topology, the convergence time of BGP and SAV mechanisms, and the SAV table generation time. The AS-level network topology with various numbers of ASes is derived from CAIDA AS topology as we analyzed in §5.1.

The experiment completion time varies for running different SAV mechanisms. Taking DSAV and E-DSAV as examples, Figure 5 shows the total experiment time of their implementations with the number of AS varying from 25 to 200. Along with the increase of the number of ASes, both the experiment completion times for DSAV and E-DSAV increase. Compared with DSAV, E-DSAV shows a slower growth trend with the increase of network size. It is because E-DSAV has less communication overhead and converges faster than DSAV.

6 CONCLUSION

In this paper, we propose a unified framework UniSAV, which provides effective support for the design, implementation, and evaluation of existing and future SAV mechanisms.

ACKNOWLEDGMENTS

We thank our reviewers for their thoughtful comments. Libin Liu is the corresponding author. This work is supported by the National Key Research and Development Program of China under Grant 2022YFB3105200.

REFERENCES

[1] [n. d.]. CAIDA AS Relationships. https://catalog.caida.org/dataset/as_relationships_serial_1. ([n. d.]).

[2] [n. d.]. Source Address Validation in Intra-domain and Inter-domain Networks (savnet). https://datatracker.ietf.org/wg/savnet/about/. ([n. d.]).

[3] 2014. Mutually Agreed Norms for Routing Security. https://www.manrs.org/. (2014).

[4] 2015. Addressing the Challenge of IP Spoofing. https://www.internetsociety.org/resources/doc/2015/addressing-the-challenge-of-ip-spoofing/. (2015).

[5] 2015. Addressing the Challenge of IP Spoofing. https://www.internetsociety.org/resources/doc/2015/addressing-the-challenge-of-ip-spoofing/. (2015).

[6] 2018. GitHub's DDoS Incident Report. https://github.blog/2018-03-01-ddos-incident-report/. (2018).

[7] 2019. MANRS Implementation Guide: Anti-Spoofing - Preventing Traffic with Spoofed Source IP Addresses. https://www.manrs.org/netops/guide/antispoofing/. (2019).

[8] 2019. NET SCOUT THREAT INTELLIGENCE REPORT 2019. https://www.netscout.com/sites/default/files/2020-02/SECR_001_EN-2001_Web.pdf. (2019).

[9] 2020. Amazon 'thwarts largest ever DDoS cyber-attack'. https://www.bbc.co.uk/news/technology-53093611. (2020).

[10] 2022. NET SCOUT DDoS THREAT INTELLIGENCE REPORT 2019. https://www.netscout.com/threatreport/ddos-threat-intelligence-report/. (2022).

[11] 2023. Krill. https://github.com/NLnetLabs/krill. (2023).

[12] 2023. P4 Open Source Programming Language. https://opennetworking.org/p4/. (2023).

[13] 2023. The BIRD Internet Routing Daemon. https://bird.network.cz/. (2023).

[14] Alexander Azimov, Eugene Uskov, Randy Bush, Job Snijders, Russ Housley, and Ben Maddison. 2023. A Profile for Autonomous System Provider Authorization. https://datatracker.ietf.org/doc/draft-ietf-sidrops-aspa-profile/. (2023).

[15] Fred Baker and Pekka Savola. 2004. Ingress Filtering for Multihomed Networks. RFC 3704, Internet Engineering Task Force. (2004).

[16] Robert Beverly and Steven Bauer. 2005. The Spoofer project: Inferring the extent of source address filtering on the Internet. In *Usenix Sruti*, Vol. 5. 53–59.

[17] Casey Deccio, Alden Hilton, Michael Briggs, Trevin Avery, and Robert Richardson. 2020. Behind closed doors: a network tale of spoofing, intrusion, and false DNS security. In *Proceedings of the ACM Internet Measurement Conference*. 65–77.

[18] P. Ferguson and D. Senie. 2000. Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing. RFC 2827, Internet Engineering Task Force. (2000).

[19] Lixin Gao. 2001. On inferring autonomous system relationships in the Internet. *IEEE/ACM Transactions on networking* 9, 6 (2001), 733–745.

[20] Martin Hellman. 1976. New Directions in Cryptography. *IEEE Transactions on Information Theory* 22, 6 (1976), 644–654.

[21] Marc Kührer, Thomas Hupperich, Christian Rossow, and Thorsten Holz. 2014. Exit from Hell? Reducing the Impact of {Amplification}{DDoS} Attacks. In *23rd USENIX security symposium (USENIX security 14)*. 111–125.

[22] Adam Langley, Alistair Riddoch, Alyssa Wilk, Antonio Vicente, Charles Krasic, Dan Zhang, Fan Yang, Fedor Kouranov, Ian Swett, Janardhan Iyengar, et al. 2017. The QUIC Transport Protocol: Design and Internet-Scale Deployment. In *Proc. ACM SIGCOMM*.

[23] Markus Legner, Tobias Klenze, Marc Wyss, Christoph Sprenger, and Adrian Perrig. 2020. EPIC: Every Packet Is Checked in the Data Plane of a Path-Aware Internet. In *Proc. USENIX Security*.

[24] Matt Lepinski, Derrick Kong, and Stephen Kent. 2020. RFC 6482: A Profile for Route Origin Authorizations (ROAs). https://datatracker.ietf.org/doc/rfc6482/. (2020).

[25] Dan Li, Jianping Wu, Mingqing Huang, Lancheng Qin, and Nan Geng. 2022. DSAV Framework: Validating Source Addresses via SAV Tables Generated by a Distributed Control-plane Protocol. https://datatracker.ietf.org/doc/slides-113-savnet-dsav-framework/. (2022).

[26] Jun Li, Jelena Mirkovic, Mengqiu Wang, Peter Reiher, and Lixia Zhang. 2002. SAVE: Source Address Validity Enforcement Protocol. In *Proc. IEEE INFOCOM*.

[27] Xin Liu, Ang Li, Xiaowei Yang, and David Wetherall. 2008. Passport: Secure and Adoptable Source Authentication. In *USENIX NSDI*.

[28] Qasim Lone, Alisa Frik, Matthew Luckie, Maciej Korczyński, Michel van Eeten, and Carlos Ganán. 2022. Deployment of Source Address Validation by Network Operators: a Randomized Control Trial. In *IEEE SP*.

[29] D McPherson, F Baker, and J Halpern. 2013. Source Address Validation Improvement (SAVI) Threat Scope. (2013).

[30] Long Pan, Jiahai Yang, Lin He, Zhiliang Wang, Leyao Nie, Guanglei Song, and Yaozhong Liu. 2022. Your Router is My Prober: Measuring IPv6 Networks via ICMP Rate Limiting Side Channels. *arXiv preprint arXiv:2210.13088* (2022).

[31] Gregor N Purdy. 2004. *Linux iptables Pocket Reference: Firewalls, NAT & Accounting*. O'Reilly Media, Inc.

[32] Lancheng Qin, Li Chen, Dan Li, Honglin Ye, and Yutian Wang. 2024. Understanding Route Origin Validation (ROV) Deployment in the Real World and Why MANRS Action 1 Is Not Followed. In *NDSS*.

[33] Lancheng Qin, Dan Li, Ruifeng Li, and Kang Wang. 2022. Themis: Accelerating the Detection of Route Origin Hijacking by Distinguishing Legitimate and Illegitimate MOAS. In *31st USENIX Security Symposium (USENIX Security 22)*. USENIX Association, Boston, MA, 4509–4524. https://www.usenix.org/conference/usenixsecurity22/presentation/qin

[34] Benjamin Rothenberger, Dominik Roos, Markus Legner, and Adrian Perrig. 2020. PISKES: Pragmatic Internet-scale Key-establishment System. In *Proc. ACM Asia CCS*.

[35] Kotikalapudi Sriram, Igor Lubashev, and Doug Montgomery. 2020. Source Address Validation Using BGP UPDATEs, ASPA, and ROA (BAR-SAV). https://datatracker.ietf.org/doc/draft-sriram-sidrops-bar-sav/. (2020).

[36] Kotikalapudi Sriram, Doug Montgomery, and Jeffrey Haas. 2020. Enhanced Feasible-Path Unicast Reverse Path Forwarding. RFC 8704, Internet Engineering Task Force. (2020).

[37] Jianping Wu, Dan Li, Libin Liu, Mingqing Huang, and Kotikalapudi Sriram. 2023. Source Address Validation in Inter-domain Networks Gap Analysis, Problem Statement, and Requirements. https://datatracker.ietf.org/doc/draft-ietf-savnet-inter-domain-problem-statement/. (2023).

[38] Su Yingying, Li Dan, Chen Li, Li Qi, and Ling Sitong. [n. d.]. dRR: A Decentralized, Scalable, and Auditable Architecture for RPKI Repository.. In *NDSS*.