

Academic work on path verification protocols

Design options,
associated overhead and performance properties

NASR side meeting

Antoine FRESSANCOURT

Path verification in academic work

Design dimensions of major proposals

- Design dimensions:
 - Mutual trust between the source and the destination?
 - Verification of the path:
 - By the destination node?
 - By every intermediate nodes?
 - Authentication of the source:
 - By the destination node?
 - By each intermediate node?
 - Probabilistic or deterministic verification scheme?
 - Encryption of routing directive?

➔ Potential impact:

- Encryption load
- Header overhead
- Processing time in routers / network nodes



How to:

- ➔ Verify the path as accurately as possible?
- ➔ Minimize overhead?
- ➔ Reduce the (cryptographic) load as much as possible?



Path verification in academic work

Comparison table of major proposals

	Scheme characteristics				Header size (bytes) / intermediate nodes			Comparison / Min. header size	
	Mutual Source-dest. Trust	Validating node (Src / Dst / Interim)	Source authent. (Dst / Interim)	Deterministic / probabilist	n	4	2	4	2
ICING	No	Yes / Yes / Yes	Yes / Yes	Deterministic	79 + 42.n	247	163	10,29	6,79
Path trace	Yes / No	No / Yes / Partial	No / No	Deterministic	48	48	48	2,00	2,00
Src Auth. By Dst and Int nodes	Yes	No / No / No	Yes / Yes	Deterministic	48 + 16.n	112	80	4,67	3,33
OPT with Trust	Yes	Yes / Yes / Yes	Yes / Yes	Deterministic	68 + 16.n	132	100	5,50	4,17
Trustless OPT	No	Yes / Yes / Yes	Yes / Yes	Deterministic	84 + 16.n	148	116	6,17	4,83
OSV	Yes	Yes / Yes / Yes	Yes / Yes	Deterministic	108 + 2.n	116	112	4,83	4,67
PPV - IPv4	Yes	No / Yes / Partial	No / No	Probabilistic	64	64	64	2,67	2,67
PPV - IPv6	Yes	No / Yes / Partial	No / No	Probabilistic	88	88	88	3,67	3,67
SRv6 HMAC*	Yes	Integrity / Int. / Int.	No / No	Passive verification	40	40	40	1,67	1,67
RFL	Yes / No	Yes / Yes / Yes	Yes / Yes	Deterministic	38 + 4.n	54	46	2,25	1,92
MASK	Yes	Yes / Yes / Partial	Yes / Partial	Probabilistic	24	24	24	1,00	1,00
EPIC - L3	Yes	Yes / Yes / No	Yes / Yes	Deterministic	36 + 5.n	56	46	2,33	1,92
SR-TPP	Yes	No / Yes / Yes	Yes / Yes	Deterministic	48 + 16.n	112	80	4,67	3,33

Path verification in academic work

Comparison table of major proposals

	Scheme characteristics				Header size (bytes) / intermediate nodes			Comparison / Min. header size	
	Mutual Source-dest. Trust	Validating node (Src / Dst / Interim)	Source authent. (Dst / Interim)	Deterministic / probabilist	n	4	2	4	2
ICING	No	Yes / Yes / Yes	Yes / Yes	Deterministic	79 + 42.n	247	163	10,29	6,79
Path trace	Yes / No	No / Yes / Partial	No / No	Deterministic	48	48	48	2,00	2,00
Src Auth. By Dst and Int nodes	Yes	No / No / No	Yes / Yes	Deterministic	48 + 16.n	112	80	4,67	3,33
OPT with Trust	Yes	Yes / Yes / Yes	Yes / Yes	Deterministic	68 + 16.n	132	100	5,50	4,17
Trustless OPT	No	Yes / Yes / Yes	Yes / Yes	Deterministic	84 + 16.n	148	116	6,17	4,83
OSV	Yes	Yes / Yes / Yes	Yes / Yes	Deterministic	108 + 2.n	116	112	4,83	4,67
PPV - IPv4	Yes	No / Yes / Partial	No / No	Probabilistic	64	64	64	2,67	2,67
PPV - IPv6	Yes	No / Yes / Partial	No / No	Probabilistic	88	88	88	3,67	3,67
SRv6 HMAC*	Yes	Integrity / Int. / Int.	No / No	-	40	40	40	1,67	1,67
Recent internet drafts		Yes / Yes / Yes	Yes / Yes	Deterministic	38 + 4.n	54	46	2,25	1,92
		Yes / Yes / Partial	Yes / Partial	Probabilistic	24	24	24	1,00	1,00
EPIC - L3		Yes / Yes / No	Yes / Yes	Deterministic	36 + 5.n	56	46	2,33	1,92
SR-TPP	Yes	No / Yes / Yes	Yes / Yes	Deterministic	48 + 16.n	112	80	4,67	3,33
SRv6-PoT	Yes	No / Yes / No	Yes / No	Deterministic	56	56	56	2,33	2,33

* Linux implementation using SHA-256

Path verification in academic work

Comparison table of major proposals

	Scheme characteristics							Comparison / Min. header size	
	Mutual Source-dest. Trust	Validating node (Src / Dst / Interim)	Source authent. (Dst / Interim)					4	2
ICING	No	Yes / Yes / Yes	Yes / Yes				163	10,29	6,79
Path trace	Yes / No	No / Yes / Partial	No / No				48	2,00	2,00
Src Auth. By Dst and Int nodes	Yes	No / No / No	Yes / Yes				80	4,67	3,33
OPT with Trust	Yes	Yes / Yes / Yes	Yes / Yes				100	5,50	4,17
Trustless OPT	No	Yes / Yes / Yes	Yes / Yes	Deterministic	$84 + 16.n$	148	116	6,17	4,83
OSV	Yes	Yes / Yes / Yes	Yes / Yes	Deterministic	$108 + 2.n$	116	112	4,83	4,67
PPV - IPv4	Yes	No / Yes / Partial	No / No	Probabilistic	64	64	64	2,67	2,67
PPV - IPv6	Yes	No / Yes / Partial	No / No	Probabilistic	88	88	88	3,67	3,67
SRv6 HMAC*	Yes	Integrity / Int. / Int.	No / No	-	40	40	40	1,67	1,67
RFL	Yes / No	Yes / Yes / Yes	Yes / Yes	Deterministic	$38 + 4.n$	54	46	2,25	1,92
MASK	Yes	Yes / Yes / Partial	Yes / Partial	Probabilistic	24	24	24	1,00	1,00
EPIC - L3	Yes	Yes / Yes / No	Yes / Yes	Deterministic	$36 + 5.n$	56	46	2,33	1,92
SR-TPP	Yes	No / Yes / Yes	Yes / Yes	Deterministic	$48 + 16.n$	112	80	4,67	3,33
SRv6-PoT	Yes	No / Yes / No	Yes / No	Deterministic	56	56	56	2,33	2,33

In these columns, we compare the path verification header's overhead compared with the path verification header of minimum size for a path with n nodes

* Linux implementation using SHA-256

Path verification in academic work

Comparison table of major proposals

	Scheme characteristics				Header size intermediate nodes				
	Mutual Source-dest. Trust	Validating node (Src / Dst / Interim)	Source authent. (Dst / Interim)	Deterministic / probabilist	n	4			
ICING	No	Yes / Yes / Yes	Yes / Yes	Deterministic	79 + 42.n	24			
Path trace	Yes / No	No / Yes / Partial	No / No	Deterministic	48	48	48	2,00	2,00
Src Auth. By Dst and Int nodes	Yes	No / No / No	Yes / Yes	Deterministic	48 + 16.n	112	80	4,67	3,33
OPT with Trust	Yes	Yes / Yes / Yes	Yes / Yes	Deterministic	68 + 16.n	132	100	5,50	4,17
Trustless OPT	No	Yes / Yes / Yes	Yes / Yes	Deterministic	84 + 16.n	148	116	6,17	4,83
OSV	Yes	Yes / Yes / Yes	Yes / Yes	Deterministic	108 + 2.n	116	112	4,83	4,67
PPV - IPv4	Yes	No / Yes / Partial	No / No	Probabilistic	64	64	64	2,67	2,67
PPV - IPv6	Yes	No / Yes / Partial	No / No	Probabilistic	88	88	88	3,67	3,67
SRv6 HMAC*	Yes	Integrity / Int. / Int.	No / No	-	40	40	40	1,67	1,67
RFL	Yes / No	Yes / Yes / Yes	Yes / Yes	Deterministic	38 + 4.n	54	46	2,25	1,92
MASK	Yes	Yes / Yes / Partial	Yes / Partial	Probabilistic	24	24	24	1,00	1,00
EPIC - L3	Yes	Yes / Yes / No	Yes / Yes	Deterministic	36 + 5.n	56	46	2,33	1,92
SR-TPP	Yes	No / Yes / Yes	Yes / Yes	Deterministic	48 + 16.n	112	80	4,67	3,33
SRv6-PoT	Yes	No / Yes / No	Yes / No	Deterministic	56	56	56	2,33	2,33

In general, the more security features, the less trust among nodes, the larger the overhead

* Linux implementation using SHA-256

Path verification in academic work

Comparison table of major proposals

	Scheme characteristics				Header size intermedia				
	Mutual Source-dest. Trust	Validating node (Src / Dst / Interim)	Source authent. (Dst / Interim)	Deterministic / probabilist	n				
ICING	No	Yes / Yes / Yes	Yes / Yes	Deterministic	$79 + 42.n$				
Path trace	Yes / No	No / Yes / Partial	No / No	Deterministic	48	48			
Src Auth. By Dst and Int nodes	Yes	No / No / No	Yes / Yes	Deterministic	$48 + 16.n$	112			
OPT with Trust	Yes	Yes / Yes / Yes	Yes / Yes	Deterministic	$68 + 16.n$	132			
Trustless OPT	No	Yes / Yes / Yes	Yes / Yes	Deterministic	$84 + 16.n$	148			
OSV	Yes	Yes / Yes / Yes	Yes / Yes	Deterministic	$108 + 2.n$	116			
PPV - IPv4	Yes	No / Yes / Partial	No / No	Probabilistic	64	64			
PPV - IPv6	Yes	No / Yes / Partial	No / No	Probabilistic	88	88	88	3,67	3,67
SRv6 HMAC*	Yes	Integrity / Int. / Int.	No / No	-	40	40	40	1,67	1,67
RFL	Yes / No	Yes / Yes / Yes	Yes / Yes	Deterministic	$38 + 4.n$	54	46	2,25	1,92
MASK	Yes	Yes / Yes / Partial	Yes / Partial	Probabilistic	24	24	24	1,00	1,00
EPIC - L3	Yes	Yes / Yes / No	Yes / Yes	Deterministic	$36 + 5.n$	56	46	2,33	1,92
SR-TPP	Yes	No / Yes / Yes	Yes / Yes	Deterministic	$48 + 16.n$	112	80	4,67	3,33
SRv6-PoT	Yes	No / Yes / No	Yes / No	Deterministic	56	56	56	2,33	2,33

Overhead depends on number of nodes on the path if intermediate nodes verify the path or authenticate the source in a deterministic way

⇔ Presence of « intermediate » fields in the header

Path verification in academic work

Comparison table of major proposals

	Scheme characteristics				Header size intermediate nodes				
	Mutual Source-dest. Trust	Validating node (Src / Dst / Interim)	Source authent. (Dst / Interim)	Deterministic / probabilist	n	4			
ICING	No	Yes / Yes / Yes	Yes / Yes	Deterministic	79 + 42.n	247			
Path trace	Yes / No	No / Yes / Partial	No / No	Deterministic	48	48			
Src Auth. By Dst and Int nodes	Yes	No / No / No	Yes / Yes	Deterministic	48 + 16.n				
OPT with Trust	Yes	Yes / Yes / Yes	Yes / Yes	Deterministic	68 + 16.n				
Trustless OPT	No	Yes / Yes / Yes	Yes / Yes	Deterministic	84 + 16.n	148			
OSV	Yes	Yes / Yes / Yes	Yes / Yes	Deterministic	108 + 2.n	116	112	4,83	4,67
PPV - IPv4	Yes	No / Yes / Partial	No / No	Probabilistic	64	64	64	2,67	2,67
PPV - IPv6	Yes	No / Yes / Partial	No / No	Probabilistic	88	88	88	3,67	3,67
SRv6 HMAC*	Yes	Integrity / Int. / Int.	No / No	-	40	40	40	1,67	1,67
RFL	Yes / No	Yes / Yes / Yes	Yes / Yes	Deterministic	38 + 4.n	54	46	2,25	1,92
MASK	Yes	Yes / Yes / Partial	Yes / Partial	Probabilistic	24	24	24	1,00	1,00
EPIC - L3	Yes	Yes / Yes / No	Yes / Yes	Deterministic	36 + 5.n	56	46	2,33	1,92
SR-TPP	Yes	No / Yes / Yes	Yes / Yes	Deterministic	48 + 16.n	112	80	4,67	3,33
SRv6-PoT	Yes	No / Yes / No	Yes / No	Deterministic	56	56	56	2,33	2,33

Probabilistic schemes have been introduced to allow a verification by intermediate nodes while limiting the linear overhead ⇔ verification done over threads of packets / network flows

* Linux implementation using SHA-256

Path verification in academic work

Comparison table of major proposals

	Scheme characteristics				Header size intermedia				
	Mutual Source-dest. Trust	Validating node (Src / Dst / Interim)	Source authent. (Dst / Interim)	Deterministic / probabilist	n	4			
ICING	No	Yes / Yes / Yes	Yes / Yes	Deterministic	79 + 42.n	247			
Path trace	Yes / No	No / Yes / Partial	No / No	Deterministic	48	48			
Src Auth. By Dst and Int nodes	Yes	No / No / No	Yes / Yes	Deterministic	48 + 16.n	112			
OPT with Trust	Yes	Yes / Yes / Yes	Yes / Yes	Deterministic	68 + 16.n				
Trustless OPT	No	Yes / Yes / Yes	Yes / Yes	Deterministic	84 + 16.n				
OSV	Yes	Yes / Yes / Yes	Yes / Yes	Deterministic					
PPV - IPv4	Yes	No / Yes / Partial	No / No	Prob	64	64			
PPV - IPv6	Yes	No / Yes / Partial	No / No	Probabilistic	88	88	88	3,67	3,67
SRv6 HMAC*	Yes	Integrity / Int. / Int.	No / No	-	40	40	40	1,67	1,67
RFL	Yes / No	Yes / Yes / Yes	Yes / Yes	Deterministic	38 + 4.n	54	46	2,25	1,92
MASK	Yes	Yes / Yes / Partial	Yes / Partial	Probabilistic	24	24	24	1,00	1,00
EPIC - L3	Yes	Yes / Yes / No	Yes / Yes	Deterministic	36 + 5.n	56	46	2,33	1,92
SR-TPP	Yes	No / Yes / Yes	Yes / Yes	Deterministic	48 + 16.n	112	80	4,67	3,33
SRv6-PoT	Yes	No / Yes / No	Yes / No	Deterministic	56	56	56	2,33	2,33

Given their minimal overhead, we have pushed our evaluation of RFL and MASK to compare the performance penalty associated with two types of path validation schemes
(RFL → deterministic, MASK → probabilistic)

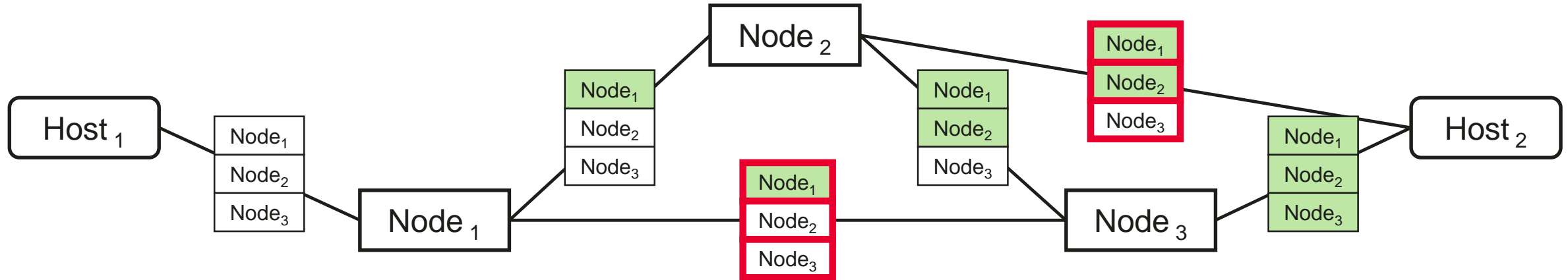
* Linux implementation using SHA-256

SRv6 HMAC Vs. RFL Vs. MASK

Attack mitigation capabilities

SRv6 HMAC Vs. RFL Vs. MASK

Diversion routing attack



- SRv6 HMAC (using SHA-256):
 - All the displayed packets carry a valid SRv6 Hash value
 - Node₁ can send packet to Node₃, (e.g. spoofing Node₂'s IP address) without Node₃ noticing anything
- With MASK (Probabilistic path verification):
 - If Node₁ tries to send the packet directly to Node₃, Node₃ can detect that Node₂ has not computed the verification hash tag
 - ➔ *Node₁'s rogue behaviour can be detected*
- With RFL (Deterministic path verification):
 - If Node₂ tries to send the packet directly to CPE₂ (e.g. spoofing Node₃'s IP address), CPE₂ may not notice a problem if the probabilistic verification occurred between Node₁ and Node₂
 - ➔ *Node₂'s rogue behaviour can be detected*

SRv6 HMAC Vs. RFL Vs. MASK

Performance overhead comparison

Evaluation of RFL and MASK compared to SRv6 HMAC

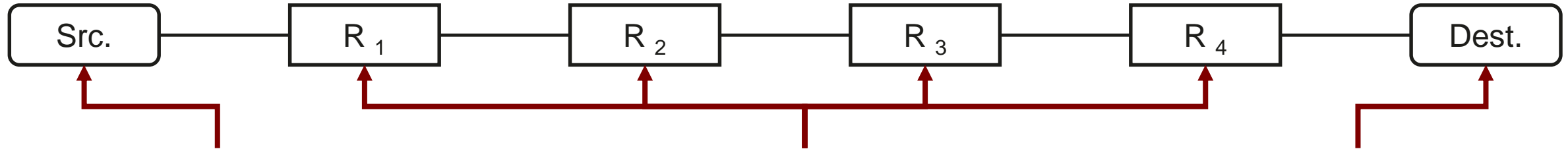
Experimental setup



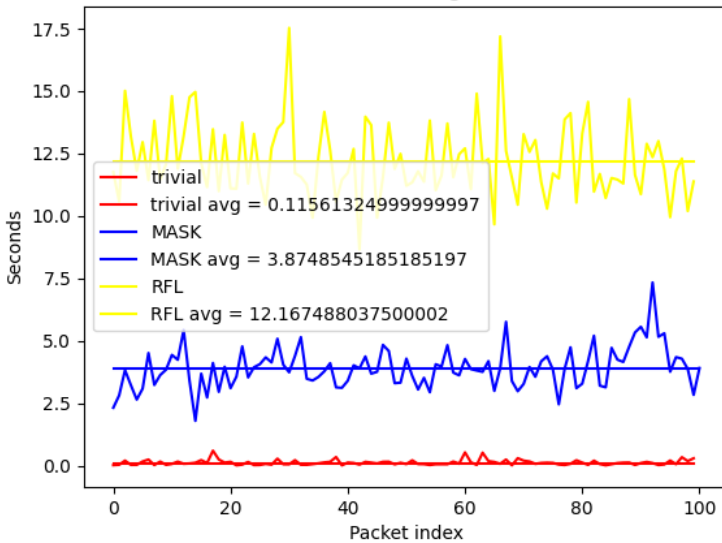
- Experimental setup in Python
- Goal: Compare MASK and RFL's performance to SRv6 HMAC (used as the baseline)
 - Performance comparison is **relative** given Python's limitations
- 3 measurements:
 1. Packet generation time at the source node (Src.)
 2. Packet processing time at the intermediate node (even if the router does not perform verification operations in a probabilistic setting)
 3. Packet verification time at the destination (Dst.)

Evaluation of RFL and MASK compared to SRv6 HMAC

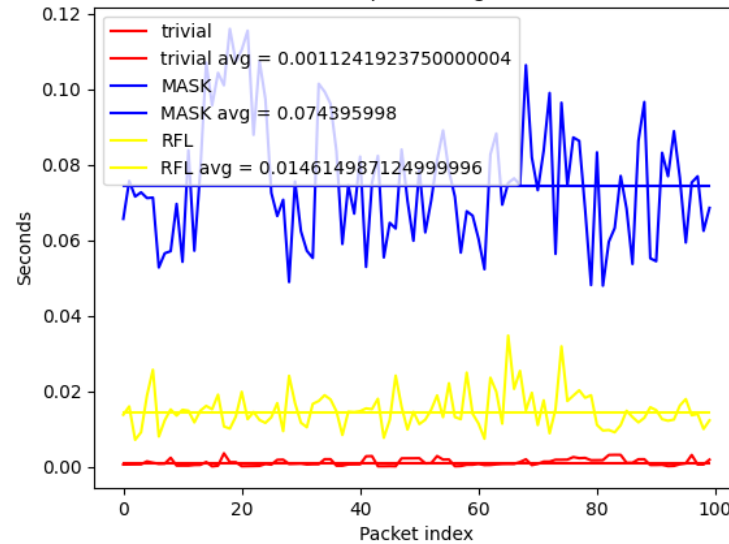
Results



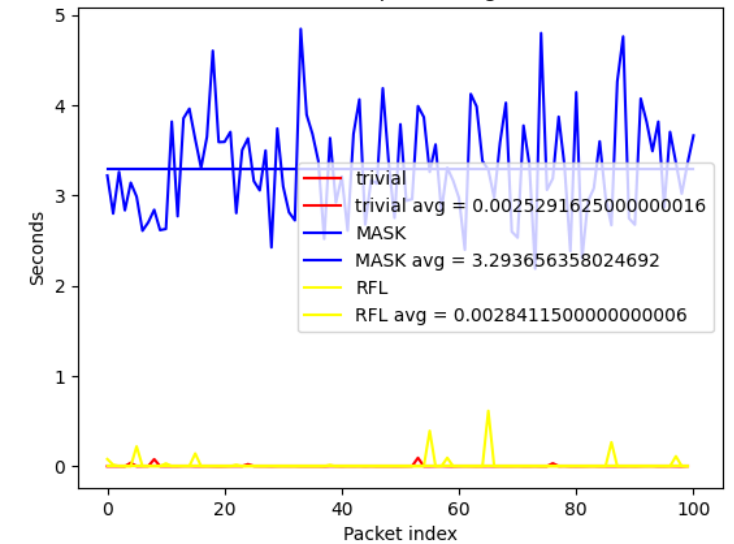
Source building time



Router processing time

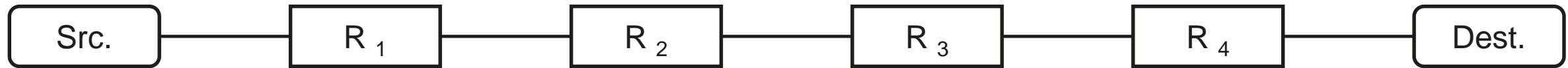


Destination processing time



Evaluation of RFL and MASK compared to SRv6 HMAC

Results



	Header size		Source processing time		Destination processing time		Intermediate node processing time	
	Size	Baseline comparison	Average measure	Baseline comparison	Average measure	Baseline comparison	Average measure	Baseline comparison
SRv6 HMAC (Baseline)	40	1	0,115613	1	0,002529	1	0,001124	1
MASK	24	0,6	3,874854	33,5157292	3,293656	1302,355081	0,074395	66,18772242
RFL	54	1,35	12,167488	105,2432512	0,002841	1,123368921	0,014614	13,00177936

- Despite a small overhead penalty, RFL performs better than MASK at intermediate nodes and at the destination on average
 - ➔ Path validation done over 5 packets rather than one in a probabilistic setting
 - ➔ Destination processing load for MASK ⇔ cost of the additional protection
- Need evaluation with implementation at lower level to be more accurate

Thank you!

Contact: Antoine Fressancourt

antoine.fressancourt@huawei.com

antoine@aft.network

Path verification in academic work

References

ICING	Naous, Jad, et al. " <i>Verifying and enforcing network paths with ICING.</i> " Proceedings of the Seventh Conference on Emerging Networking Experiments and Technologies. 2011.
Path Trace & OPT flavors	Kim, Tiffany Hyun-Jin, et al. " <i>Lightweight source authentication and path validation.</i> " Proceedings of the 2014 ACM Conference on SIGCOMM. 2014.
OSV	Cai, Hao, and Tilman Wolf. " <i>Source authentication and path validation in networks using orthogonal sequences.</i> " 2016 25th International Conference on Computer Communication and Networks (ICCCN). IEEE, 2016.
PPV	Wu, Bo, et al. " <i>Enabling efficient source and path verification via probabilistic packet marking.</i> " 2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS). IEEE, 2018.
RFL	Wu, Bo, et al. " <i>RFL: Robust fault localization on unreliable communication channels.</i> " Computer Networks 158 (2019): 158-174.
MASK	Fu, Songtao, et al. " <i>MASK: practical source and path verification based on Multi-AS-Key.</i> " IEEE/ACM Transactions on Networking (2022).
EPIC	Legner, Markus, et al. " <i>{EPIC}: every packet is checked in the data plane of a {Path-Aware} Internet.</i> " 29th USENIX Security Symposium (USENIX Security 20). 2020.
SR-TPP	Zhou, Jiang, et al. " <i>SR-TPP: Extending IPv6 segment routing to enable trusted and private network paths.</i> " 2020 IEEE Symposium on Computers and Communications (ISCC). IEEE, 2020.
SRv6-HMAC	Filsfils, Clarence, et al. " <i>IPv6 segment routing header (SRH).</i> " IETF RFC 8754, available: https://datatracker.ietf.org/doc/rfc8754/ (2020).
SRv6-PoT	Iannone, Luigi et al. " <i>Segment Routing over IPv6 (SRv6) Proof of Transit.</i> " IETF Internet draft, available: https://datatracker.ietf.org/doc/draft-iannone-spring-srv6-pot/ (2024).

Verifying and enforcing network paths with ICING

ACM CoNEXT '11

- Use of 2 key enablers:

1. Aggregate message authentication codes

- To verify a path, a given node should verify that each node before it was traversed $\rightarrow O(n^2)$ verifier fields in the header in a classic setup
- In Aggregate MACs, verification fields are XORed to reduce the number of fields to carry $\rightarrow O(n)$ verifier fields









2. Self-certifying names

- Nodes locally compute a private/public key pair $\langle k_i, N_i \rangle$, and use N_i as their name
- Symmetric keys are generated using a function f such that, for a pair of nodes i and j :




$$f(k_i, N_j) = f(N_j, k_i)$$

\rightarrow *Non-interactive key exchange primitive*

- Use of both primitives for path verification

Path	Proofs	Verifiers
R0		
R1		
R2		 
R3		 



Path	Verifiers
PK0	
PK1	
PK2	
PK3	

Verifying and enforcing network paths with ICING

Path verification in details

P	N_0	N_1	N_2	N_3
V_1	$A_1 \oplus \text{PoP}_{0,1}$			
V_2	$A_2 \oplus \text{PoP}_{0,2}$			
V_3	$A_3 \oplus \text{PoP}_{0,3}$			
	Payload			

(a) updated by N_0
to be verified by N_1

N_0	N_1	N_2	N_3
$A_1 \oplus \text{PoP}_{0,1}$			
$A_2 \oplus \text{PoP}_{0,2} \oplus \text{PoP}_{1,2}$			
$A_3 \oplus \text{PoP}_{0,3} \oplus \text{PoP}_{1,3}$			
Payload			

(b) updated by N_1
to be verified by N_2

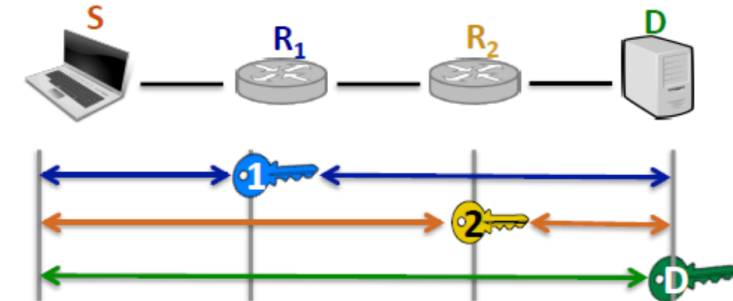
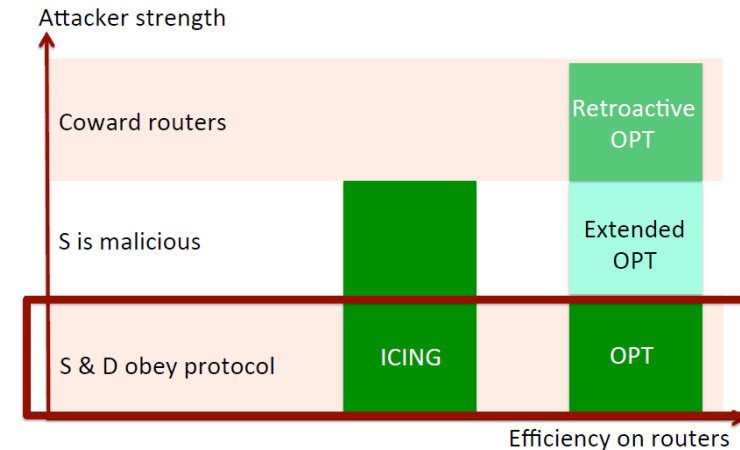
N_0	N_1	N_2	N_3
$A_1 \oplus \text{PoP}_{0,1}$			
$A_2 \oplus \text{PoP}_{0,2} \oplus \text{PoP}_{1,2}$			
$A_3 \oplus \text{PoP}_{0,3} \oplus \text{PoP}_{1,3} \oplus \text{PoP}_{2,3}$			
Payload			

(c) updated by N_2
to be verified by N_3

Origin and Path Trace (OPT)

Lightweight source authentication and path validation – ACM SIGComm '14

- Design of 3 path verification variants to enhance ICING
 - Retroactive OPT → Path verification after the packet has been received, used to analyze awkward behaviors
 - OPT → Used to verify a path when the source and destination are trusted
 - Extended-OPT → Used to verify the path in case the source is not trusted
- OPT principle:
 - S selects a path to D
 - Nodes establish shared secret key(s) with S & D
 - S prepares special fields for each node in the packet header → Helps each router derive shared key & authenticate source
 - Each node updates a verification field in the packet header → Helps downstream nodes validate path
- Advantage compared to ICING:
 - More efficient on routers: $O(1)$ verifier field instead of $O(n)$
 - Coward routers attack prevention



Orthogonal Sequence Verification (OSV)

Source Authentication and Path Validation in Net. Using Orthogonal Sequences – ICCCN '16

Path verification mechanism with 3 characteristics:

1. Low Verification Time:

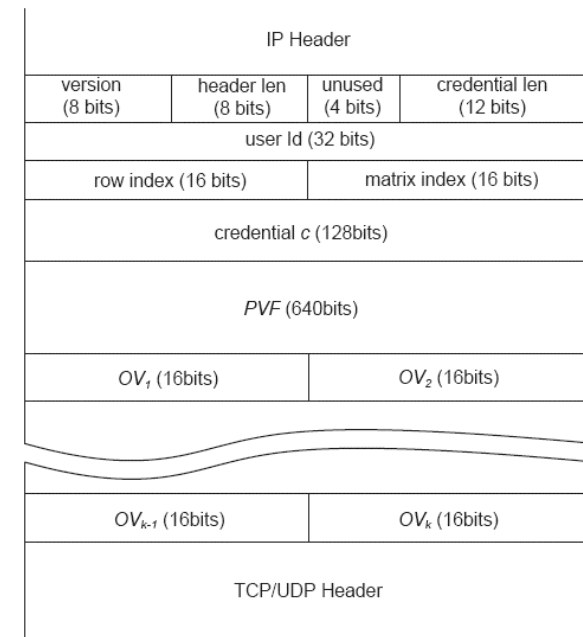
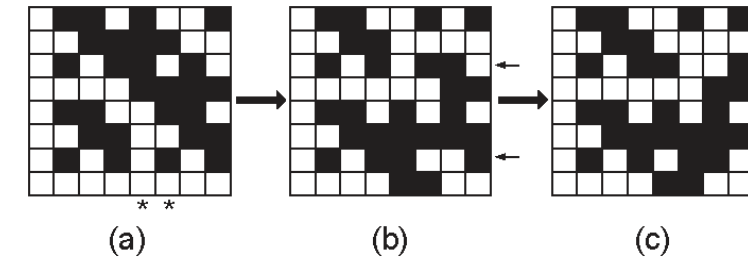
- Use of an inner product computation based on Hadamard matrixes
 - $M \times m$ matrixes such that $HH^T = mI_m$
- ➔ Replacement of complicated cryptographic operations
- ➔ Decreased verification time compared to ICING ($\sim 0.01\mu s$ Vs. $24.4\mu s$)
- ➔ Faster setup compared to OTP (0.15 ms Vs. 4 ms)

2. Low Packet Overhead:

- Relatively small OSV header (~ 125 bytes)
- ➔ Smaller header if path length > 4

3. Non-dependence on Path Length:

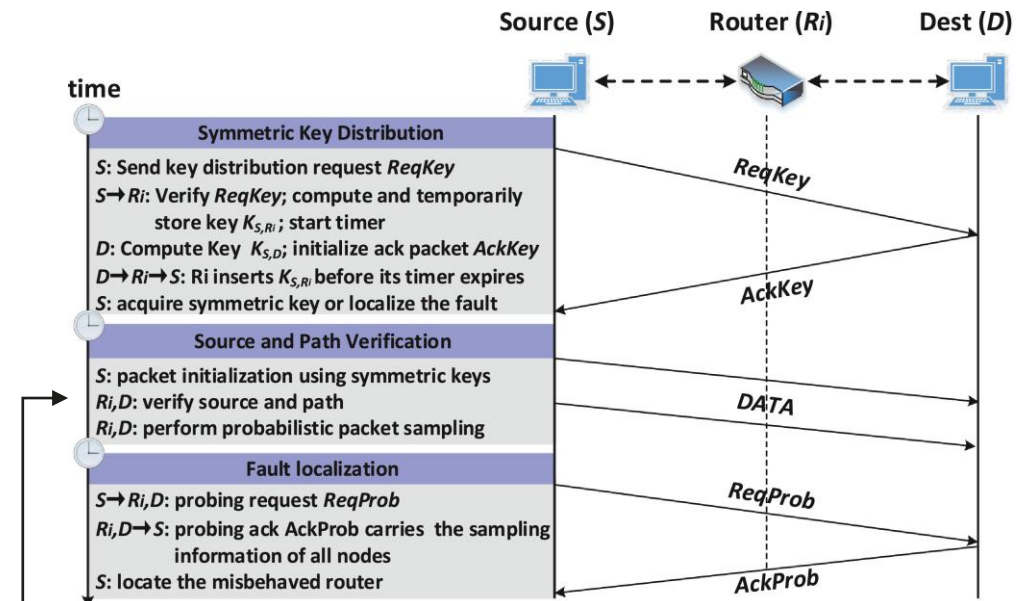
- Packet processing time in routers is almost independent of the number of hops that are traversed by the packet.
- OSV's packet header overhead increased by 2 bytes for each additional hop.



Robust Fault Localization (RFL)

RFL: Robust fault localization on unreliable communication channels

- Each packet's source is authenticated and the path is verified for each packet.
 - Setup phase using public key cryptography to allow S to retrieve symmetric keys used during an Epoch.
 - During data phase, symmetric key used by intermediate routers to verify a packet.
 - Verification uses ID of predecessor and marks computed for successor nodes (See Eq. 12)
 - Small overhead
- Additional reporting mechanism to locate router / nodes causing an issue



$$RFLheader = \{SessionID, epoch, PacketID, M_{Path}\}. \quad (9)$$

$$PacketID = H(SessionID || epoch || IP_{cst}), \quad (10)$$

$$M_{Path} = \langle M_1, M_2, \dots, M_n, M_D \rangle, \quad (11)$$

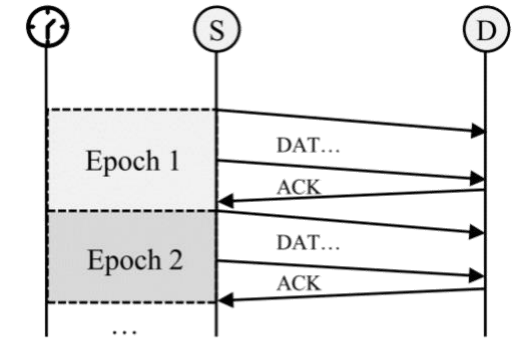
$$M_D = PRF_{K_{S,D}}(M_{cst}^{in} || TTL_D || R_n). \quad (12)$$

$$M_i = PRF_{K_{S,R_i}}(M_{cst}^{in} || TTL_i || R_{i-1} || M_{i+1} || \dots || M_n || M_D).$$

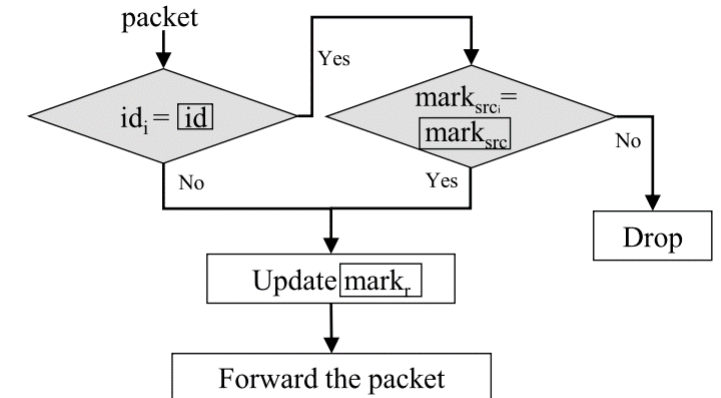
MASK

Practical Source and Path Verification Based on Multi-AS-Key

- Path verification based on a probabilistic packet marking mechanism
 - Similar as PPV by same authors
 - ➔ *Fixed header size whatever the path length*
- Source is always authenticated by destination
- Key distribution and dataplane are separated to ease key management operations
 - Targeting reconstruction efficiency rather than full security
 - Key change frequently and are distributed by a Key Distribution Server



(a) Processing at end host



(b) Processing at router