Marmot: a Light-weight Secure SRv6 Header Authentication System

Sae Hyong Park
Mobile Core Network Research Section
ETRI
Daejeon, Republic of Korea
labry@etri.re.kr

Namseok Ko
Mobile Core Network Research Section
ETRI
Daejeon, Republic of Korea
nsko@etri.re.kr

***Abstract*—As network architectures continue to evolve, ensuring security and data integrity becomes paramount. This paper presents Marmot, a novel lightweight authentication system tailored specifically for SRv6 (Segment Routing with IPv6) headers. Marmot addresses the challenges of authentication and integrity within SRv6 by introducing a unique approach that amalgamates the scalability benefits of SRv6 with the HMAC (Hash-based Message Authentication Code) framework. This fusion is realized through the strategic use of mirror ports for headend nodes, with P nodes engaging in mutual authentication. Marmot offers a triad of advantages: scalability, simplicity, and performance. By harnessing the power of mirroring ports and HashChain-based signatures, the system ensures the integrity of SRv6-based communications while maintaining the efficiency required in modern network environments.**

***Index Terms*—HashChain, SRv6, HMAC, authentication, integrity**

I. Introduction

In recent years, the rapid evolution of networking technologies has led to the development of innovative paradigms aimed at addressing the challenges posed by modern network architectures. One such paradigm is Segment Routing with IPv6 (SRv6) [1], [2], [3], [4], which offers a flexible and efficient approach to packet forwarding by allowing routers to explicitly define paths through the network using a sequence of segment identifiers (SIDs). While SRv6 brings remarkable benefits, it also introduces security concerns that demand specialized solutions to ensure the integrity and authenticity of routed packets.

Security in networking has always been a paramount concern, and SRv6's unique characteristics raise novel challenges for safeguarding data as it traverses through dynamic paths within the network. For this puporse, the integration of Hash-based Message Authentication Code (HMAC) verification [5] is integrated into SRv6. The HMAC technique provides a robust mechanism to verify the authenticity and integrity of segments added to packets, thus thwarting unauthorized modifications and enhancing the overall security posture of SRv6-enabled networks.

This paper introduces a critical breach scenario and shows how to redesign SRv6 networks while maintaining their original strengths such as scalability and flexibility. Through a comprehensive analysis of SRv6 architecture, security challenges, and the role of HMAC, this paper offers valuable insights into the pragmatic implementation and deployment of secure SRv6 networks. Figure 1a shows two segments that belong to VPN1 and VPN2. The traffic of VPN1 is routed to CE4 without any filtering, but the traffic of VPN2 needs to go through a firewall since we presume CE1 is less trusted than CE2. However, in Figure 1b, we show a case where P1 is compromised by an attacker and the VPN2 traffic is routed without going through the firewall by attaching the SID list and the HMAC of VPN1. This is possible because the current SRv6 standards only hashes the SID list and a shared key. This opens up an opportunity for a replay attack. An attacker can easily utilize the HMACs for another flow to redirect the other flows. However, we design a system that can detect this replay attacks while maintaining the scalability of SRv6. By addressing these security concerns, network operators can confidently embrace SRv6 for its benefits while maintaining the authenticity and integrity of transmitted data.

The rest of this paper is organized as follows. In Section II, we describe the overall architecture and detailed procedures of Marmot. Section III provides related work. Section IV covers the conclusion and future work on further enhancing Marmot.

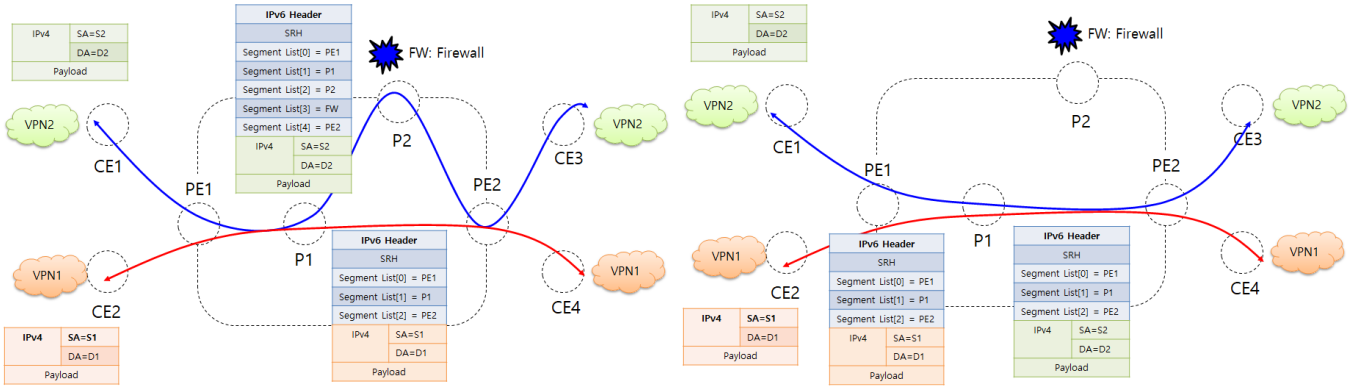
II. Marmot Design

Marmot is designed to achieve the following goals as an integrated authentication system in SRv6:

- Scalability: the primary motivation for inventing SRv6 was scalability. Therefore, it is crucial to maintain the scalability of the system while securing the HMAC mechanism in SRv6.
- Simplicity: the other pivotal motivation for SRv6 was simplicity. Marmot is simple and straightforward to adopt to an SRv6 network.
- Performance: Marmot provides a balance between the performance and the security by selecting the time period of exchanging HMAC integrity tables with each other nodes and the controller.

A. Overview

Marmot is a light-weight secure authentication method for SRv6 headers. The current SRv6 networks optionally adopt HMAC to provide authentication and integrity protection for



(a) Intended Flow

(b) Maliciously Redirected Flow

Fig. 1: a Problematic Scenario of using HMAC in SRv6

SRv6 segments. This helps ensure that the segments in an SRv6 header have not been modified by unauthorized parties during transit. By using HMAC, the sender can generate a hash value based on the SRv6 header and a secret key. The recipient can then calculate the hash value based on the received SRv6 header and the same secret key. If the calculated hash matches the received hash, it indicates that the SRv6 header has not been tampered with. However, it only hashes the SID list and shared key which leads to being vulnerable to replay attacks. Marmot is a system that prevents such attacks by creating an authentication system that comprises of HashChain Generating Agent and HMAC integrity Table and mirror ports at the headend nodes.

The first goal of Marmot design, scalability, is addressed by using the hybrid method of peer verification and controller verification in the network. The controller verification is used only for the headend nodes and the peer verifications are enforced for other nodes, thus minimizing the overhead of installing mirror ports and controller. The aim of simplicity in Marmot's design is accomplished by also utilizing the HashChain because the operators can opt to use signature-based keys since HashChain can make the cost of using signature-based mechanisms constant. To meet the third objective of the Marmot design, HashChain, a lightweight mechanism for generating and verifying signatures, is integrated into the Marmot system. [6].

B. Marmot HashChain Generating Agent

As in Figure 2, Marmot HashChain Generating Agent can run across multiple regions across networks. Marmot HashChain Generating Agent operates on the controller, PE, and P creates a HashChain using packets entering the incoming port, and in the case of the controller, a HashChain is created using the packets from the mirror port. The agent creates a HashChain, and the previous HashChain is included to create the next HashChain. HMAC or ECDSA [7] with SHA-256 may be used as the HashChain.

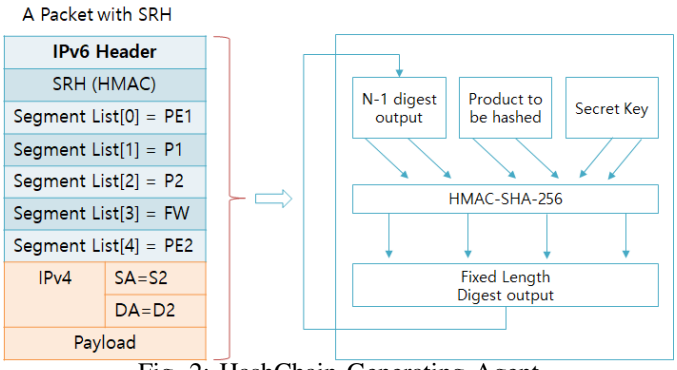


Fig. 2: HashChain Generating Agent

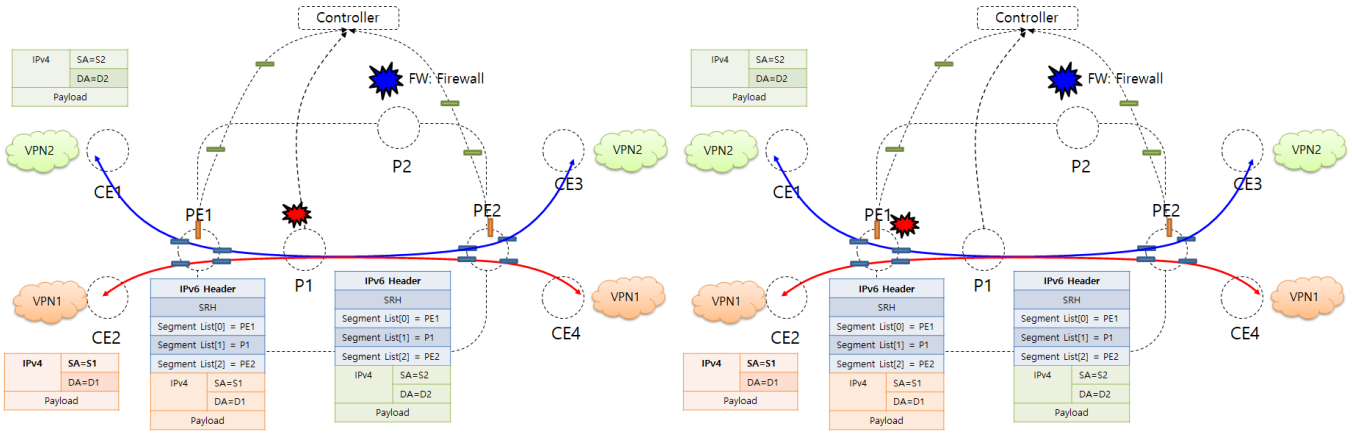
C. HMAC Integrity Table

The segment integrity table consists of a path ID, hash or HMAC, and a version of hash. The path ID consists of (headend, color, endnode) that distinguishes SRv6 policies, and (candidate path, segment list) that distinguishes segments. For the implementation, HMAC can be used, and in this case, the shared secret key should also be hashed. The generated table values are compared with each other by a set period and the operator is notified if there is a problem in integrity, which may be implemented as software based on VPP (Vector Packet Processing) [8] or P4 [9].

D. Replay Attack Detecting procedures

Upon receiving the operator's request to activate HashChain, entries for the HMAC Integiryt table are generated as follows:

① The Marmot security activation is initiated by the operator.
② Set the verification periods and HMAC Integrity Table exchange cycle for configuration.
③ Unlike RFC 8754, it generates a hash including all fields except variable fields such as TTL, Segment Left, and DA (Destination Address). As the initial hash input, a value filled with zero is used.
④ If it encounters the verification period, create a new output or return to the previous step.



(a) P1 Compromised

(b) PE1 Compromised

Fig. 3: Two cases where nodes are compromised

⑤ Check the HMAC Integrity Table exchange cycle and check the verification period again.
⑥ The values of the HMAC Integrity Table are exchanged with neighboring nodes or controllers for all P, PE, and controllers.
⑦ If the hash value is the same as the version of the HashChain, it returns to step 2, which is the step of checking the verification period.
⑧ Notify the operator if a problem is found.

Figure 3a shows the procedures of how Marmot detects the tampering of SRv6 header in P or PE. We are going to explain the case of a P compromise case first.

❶ The operator enables Secure SR through the network controller.
❷ In this step, each PE sets a HashChain generation every 100 packets (verification period), have 10-second HMAC Integrity Table exchange cycles, and 2 segment lists through the network controller.
❸ The PE delivers the configuration information to the surrounding node through an IGP extension [10] message or a dedicated message, and the surrounding node applies the configuration.
❹ PE1 in Figure 3a starts generating the HashChain according to the cycle as shown in Figure 3a. Since the initial hash value does not exist, a value filled with zeros of the same size is used. PE2 works the same as PE1. P1, P2, and network controllers operate the same as PE1.
❺ Every 10 seconds, the HMAC Integrity Table exchange cycle, PE1 exchanges and compares the network controller with P1 and tables containing the HashChain and version. PE2 exchanges the tables with network controllers and P2 and P1 with HashChains and versions, and notifies operators when they find other errors (e.g., missing entries in the segment integrity table), including mismatches in hash values.
❻ the network controller's hash chain comparison is used to detect the packet as a mirror port set in PE2 and PE1 where P1 loses its root authority and the segment to P2 goes directly to PE2 and informs the operator of the event.

Figure 3b shows the procedures of how Marmot detects the tampering of SRv6 header in PE.

❶ The operator enables Secure SR through the network controller.
❷ In this step, each PE sets HashChain generation every 100 packets (verification period), 10-second HashChain exchange cycles, and 2 segment lists through the network controller.
❸ The PE delivers the configuration information to the surrounding node through an IGP extension message or a dedicated message, and the surrounding node applies the configuration.
❹ PE1 in Figure 3b starts generating the hash chain according to the cycle as shown in Figure 3b. Since the initial hash value does not exist, a value filled with zeros of the same size is used. PE2 works the same as PE1. P1, P2, and network controllers operate the same as PE1
❺ Every 10 seconds, the hash chain exchange cycle, PE1 exchanges and compares the network controller with P1 and tables containing the hash chain and version. PE2 exchanges tables with network controllers and P2 and P1 with HashChains and versions, and notifies operators when they find other errors (e.g., missing entries in the segment integrity table), including mismatches in hash values.
❻ In Figure 7, the network controller receives the packet with the mirror port set in PE1 where PE1 loses its root authority and the segment to P2 goes directly to PE2 and detects it through a HashChain comparison between PE2 and informs the operator of the event. At this time, PE1 can find out by PE2 when sending a HashChain before the packet is tempered, and of course, if PE1 sends a tempered value, it can also be found in contrast to PE1.

III. Related Work

A. On SRv6 Security

SRv6 functions as a routing framework that facilitates a cooperative dynamic between a central network controller and network nodes. In this arrangement, IPv6 routers manage multi-hop ECMP-aware segments, while the controller, overseeing Traffic Engineering policies, integrates these segments into a source-routed pathway across the network. The significance of this system hinges on the Segment Routing Header (SRH) [11], a specific packet header containing flow state information determined at the network entry point. With the growing success and widespread adoption of such methodologies and technologies, this study presents an overview and addresses potential concerns regarding manipulation of the Segment Routing Header content. Furthermore, Bascio et al [12] outline the specifics of an experimental test setup designed to assess these identified issues.

B. Security Considerations for SRv6 Networks

Security concerns in SRv6 [13] arise primarily from vulnerabilities within SIDs (Segment IDs) and SRH (Segment Routing Header), which can be exploited much like in traditional source routing. SIDs and SRH, when unprotected or exposed to external elements, could be manipulated to initiate various types of attacks. Attackers can intercept, modify, falsify, or misuse SIDs and SRH, leading to the following threats:

- DoS/DDoS attacks: Attackers could construct segment lists that cause packets to loop between routers or hosts on specific links. ICMPv6 error messages might also be exploited to attempt DoS/DDoS attacks by sending error-inducing destination addresses or SRH in consecutive packets.
- Escaping security checks: Crafted segment lists can enable malicious packets to bypass firewall devices or reach otherwise inaccessible Internet systems.
- Traffic interception: Unprotected SIDs can be exploited to intercept traffic, potentially facilitating man-in-the-middle attacks.
- Topology discovery: SRH leaks network information, including topology, traffic flows, and service usage. However, this disclosure is less relevant to SR due to attackers having alternative means of gathering such information.
- Identity spoofing: Attackers can masquerade as authorized hosts, using their identity to access authorized services.

Additionally, SRv6 inherits security vulnerabilities from IPv6. This includes threats such as eavesdropping on service data, packet falsification, identity spoofing, packet replay, and DoS/DDoS attacks [14]. Many of these risks are not exclusive to SRv6 and are present in IPv6 and IPv4 networks as well. A comprehensive analysis of IPv6 security considerations can be found in RFC9099 [15]. Therefore, addressing these vulnerabilities is vital not only for SRv6 but for network security in general.

IV. Conclusion and Future Work

In this study, we introduce Marmot, a lightweight and secure authentication system tailored for SRv6 networks. Marmot's core concept involves integrating the scalability aspect of SRv6 into HMAC. This is achieved through the utilization of mirror ports exclusively for headend nodes, with P nodes mutually authenticating each other. Marmot offers scalability, simplicity, and performance benefits, enabled by features like the innovative use of mirroring ports and a HashChain-based signature mechanism.

In forthcoming endeavors, our plan involves the practical implementation of Marmot, accompanied by comprehensive experiments aimed at demonstrating its scalability and performance. The implementation of Marmot will be made publicly available, fostering further exploration and research in the realm of SRv6 authentication and data integrity.

V. Acknowledgment

This work was supported by Electronics and Telecommunications Research Institute (ETRI) grant funded by the Korean government [23ZR1100, A study of hyper-connected thinking Internet technology by autonomous connecting, controlling and evolving ways].

References

[1] C. Filsfils, P. Camarillo, J. Leddy, D. Voyer, S. Matsushima, and Z. Li, "Srv6 network programming," *Internet-Draft*, 2017.

[2] J. Leddy, D. Voyer, S. Matsushima, and Z. Li, "Rfc 8986: Segment routing over ipv6 (srv6) network programming," 2021.

[3] C. Filsfils, S. Previdi, L. Ginsberg, B. Decraene, S. Litkowski, and R. Shakir, "Segment routing architecture," Tech. Rep., 2018.

[4] P. L. Ventre, S. Salsano, M. Polverini, A. Cianfrani, A. Abdelsalam, C. Filsfils, P. Camarillo, and F. Clad, "Segment routing: a comprehensive survey of research activities, standardization efforts, and implementation results," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 1, pp. 182–221, 2020.

[5] H. Krawczyk, M. Bellare, and R. Canetti, "Hmac: Keyed-hashing for message authentication," Tech. Rep., 1997.

[6] T. Refaei, M. Horvath, M. Schumaker, and C. Hager, "Data authentication for ndn using hash chains," in *2015 IEEE Symposium on Computers and Communication (ISCC)*. IEEE, 2015, pp. 982–987.

[7] D. Johnson, A. Menezes, and S. Vanstone, "The elliptic curve digital signature algorithm (ecdsa)," *International journal of information security*, vol. 1, pp. 36–63, 2001.

[8] D. Barach, L. Linguaglossa, D. Marion, P. Pfister, S. Pontarelli, and D. Rossi, "High-speed software data plane via vectorized packet processing," *IEEE Communications Magazine*, vol. 56, no. 12, pp. 97–103, 2018.

[9] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese *et al.*, "P4: Programming protocol-independent packet processors," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 87–95, 2014.

[10] C. Filsfils, A. Bashandy, B. Decraene, and Z. Hu, "Rfc 9352: Is-is extensions to support segment routing over the ipv6 data plane," 2023.

[11] S. Previdi, J. Leddy, S. Matsushima, and D. Voyer, "Rfc 8754: Ipv6 segment routing header (srh)," 2020.

[12] D. L. Bascio and F. Lombardi, "On srv6 security," *Procedia Computer Science*, vol. 201, pp. 406–412, 2022.

[13] C. Li, Z. Li, C. Xie, H. Tian, and J. Mao, "Security considerations for srv6 networks," *Internet Engineering Task Force, Internet-Draft draft-li-spring-srv6-security-consideration-07*, 2021.

[14] J. Mirkovic and P. Reiher, "A taxonomy of ddos attack and ddos defense mechanisms," *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 2, pp. 39–53, 2004.

[15] É. Vyncke, K. Chittimaneni, M. Kaeo, and E. Rey, "Rfc 9099: Operational security considerations for ipv6 networks," 2021.