Canary: a Scalable Content Integrity Verifying Protocol for ICN

Yong Yoon Shin, Sae Hyong Park, Quang Tung Thai, and Sung Hyuk Byun
ETRI
Daejeon, Korea
{uni2u,labry,tqtung,shbyun}@etri.re.kr

ABSTRACT

The per-packet signature mechanism in NDN is a basic mechanism to provide in-network security. Consumers can validate provenance and integrity with the public key-based signature attached with each Data packet. However, the creation and validation processes of signature cause significant performance bottlenecks in both of consumers and producers. The embedded manifest mechanism was proposed to ease the signing overhead for streaming data producers; a signed manifest packet being composed of digests of subsequent Data packets is inserted per bundle of Data packet while each Data packet has only its digest as SignatureInfo. For a large file, the embedded manifest mechanism still needs producers to sign multiple manifest packets. The basic idea of proposed mechanism, Canary, is to enable per-segment provenance and data integrity validation with only one signing operation of producers even for a large file by exploiting the properties of Merkle tree.

CCS CONCEPTS

• **Networks** → *Network design principles*; *Network protocol design.*

KEYWORDS

NDN, integrity, provenance, validation

ACM Reference Format:
Yong Yoon Shin, Sae Hyong Park, Quang Tung Thai, and Sung Hyuk Byun. 2019. Canary: a Scalable Content Integrity Verifying Protocol for ICN. In *6th ACM Conference on Information-Centric Networking (ICN '19), September 24–26, 2019, Macao, China.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3357150.3357418

1 INTRODUCTION

In the current NDN protocol [1], it adopts a per-packet signature scheme to validate Data packet by default. However, signing and validation of each packet with a signature has performance issues. Even the state-of-the-art servers equipped with two Intel 8-core Xeon processors can generate up to about 6,000 RSA signatures per second [2]. To ease the signing overhead, the embedded manifest approach [3] has been proposed.

ICN '19, September 24–26, 2019, Macao, China

ACM ISBN 978-1-4503-6970-1/19/09…$15.00
https://doi.org/10.1145/3357150.3357418

The approach is based on embedded manifests which have digests of subsequent Data packets. According to [3], only the manifest packet itself is signed by producers and other Data packets have only digests of themselves. An example of content segmentation with embedded manifest:

/a/b/0 - manifest segment containing /a/b/1, /a/b/2, a/b/3 names with digests
/a/b/1 - data segment
/a/b/2 - data segment
/a/b/3 - data segment
/a/b/4 - manifest segment containing /a/b/5, /a/b/6, a/b/7 names with digests
/a/b/5 - data segment
...

This approach has an advantage that provenance and integrity of each data segment can be verified with only its digest and signed manifest containing the digest. When using a segment size of 1,500B and the file is a video of 1.5GB, then it requires 50,000 signing operations when assuming 20 message digests in a manifest packet. This is a significant improvement over the original NDN approach which requires 1M per-packet signing operations. This approach cuts down the overhead of signing operation by $1/k$ when there are k digests in a manifest. It has achieved notable performance enhancements in general. However, it still needs n/k signing operations when there are n segments in a data file.

This paper proposes a novel per-segment provenance and integrity verification mechanism, Canary, with only a single signing operation independently of data file size. Canary provides provenance and integrity validation with a single signature regardless of the number of chunks in a file, which utilizes Merkle tree [5]. Canary reduces greatly the number of signing operations of producers to only one, which is a significant improvement compared to n signing operations of original NDN per-segment signing and n/k signing operations in embedded manifest mechanism.

2 CANARY DESIGN

In Canary, a Merkle tree is generated using the whole sequence of Data segments of a data file. A producer only signs the root hash of the Merkle tree and sends it as the first Data packet. Each Data segment has a list of hash nodes of Merkle tree which are required to verify itself with the signed root hash. A new type of SignatureInfo is defined for the list of hash nodes.

As in Figure 1, when a consumer requests a data file, a producer responds with a Canary manifest packet containing signed root hash which will be used to validate subsequent Data packets in addition to a list of data segments. The number of required signing operations of producers is only one regardless of the number of segments.

Figure 2 explains per-segment provenance and integrity verification procedure for a 16-segment data file in Canary. D_k is the k^{th} Data segment, H_k is the hash of D_k, and H_{i-j} is the hash of Data

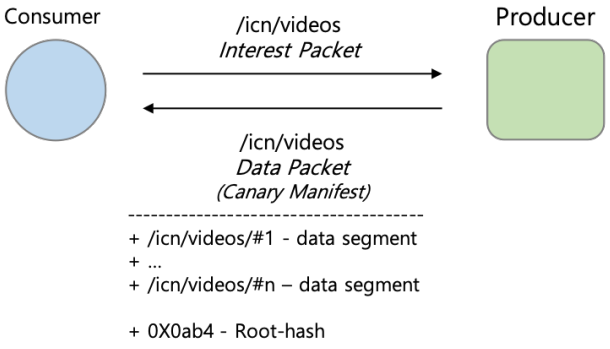

Figure 1: The overall structure of Canary

segments from D_i to D_j. A consumer receives the Canary manifest at first. Then it knows the root hash of Merkle tree and a list of of all data segments of a data file. From the list, the consumer request the first Data segment, D_1, which has Merkle tree node values, H_2, H_{3-4}, H_{5-8}, and H_{9-16}. After receiving the first segment, through calculation of H_1, H_{1-2}, H_{1-4}, H_{1-8}, and H_{1-16} which is the root hash, the consumer can verify the provenance and integrity of D_1. The second Data segment D_2 needs no Merkle tree node values because of required node values for D_2 are already calculated or received in D_1 verification stage. D_3 should be sent with H_4. The remaining Data segments can be verified with the same procedure.

Canary can be applied without any modification for dynamic contents such as live streaming video due to the buffering characteristics of real-time streaming services. The real-time streaming services such as YouTube and Netflix do buffering of its contents in the unit of about 2 to 10 second chunk before transmission to adapt to network speed variation [4]. A thousands of Data segments will be buffered and Canary can reduce the producing overhead significantly by running the same mechanism to each buffered video chunk.

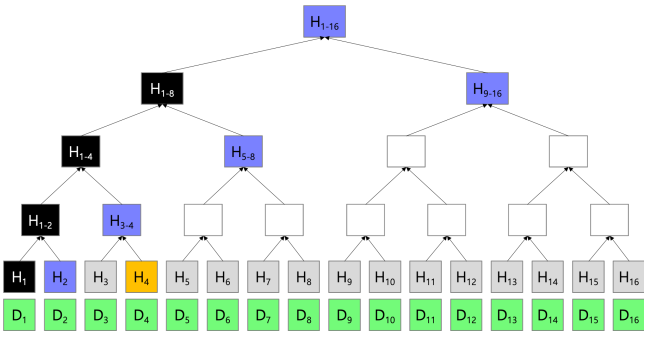

Figure 2: Analytical comparison of number of signing operations

Figure 3 shows the comparison of required signing operations of Canary compared with existing mechanism, for static data file producing and live video streaming. The data file is assumed as 1.5GB and buffered video chunk size as 3MB, and data segment size is 1,500B and each embedded manifest contains 20 digests. For a data file of n segments, native NDN producer should performs n signing operations, and embedded manifest mechanism reduces it to n/k operations, and proposed Canary reduces greatly the number of signing operations of producers to only one.

The number of signing operations in original NDN increases linearly with the increase of data file size. The embedded manifest reduced it significantly compared with the original mechanism. The proposed Canary maintains the number as one regardless of data file size. Canary reduces greatly the number of signing operations of producers to only one, which is a significant improvement compared to n signing operations of original NDN per-segment signing and n/k signing operations in embedded manifest mechanism.

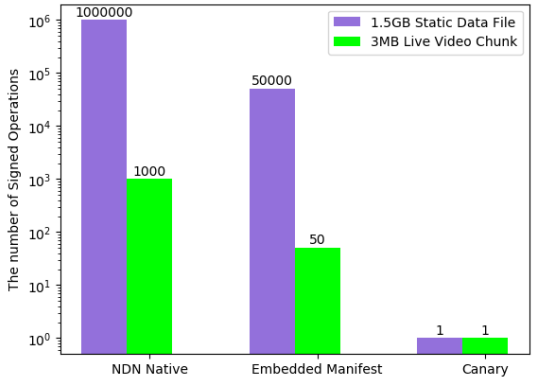

Figure 3: The Comparison of The Number of Signing Operations for a Static File and Live Video Producing

3 FUTURE WORK

The architectural benefits of Canary are promising. The key idea of Canary is to exploit the concept of Merkle tree validation into the NDN chunk validation process. As the next step, we are planning to implement Canary mechanism for consumers and producers, and perform benchmark test of end-to-end file producing and retrieval performances.

ACKNOWLEDGMENTS

This work was supported by the ICT R&D program of MSICT/IITP. [2017-0-00045, Hyper-connected Intelligent Infrastructure Technology Development]

REFERENCES

[1] Van Jacobson, Diana K Smetters, James D Thornton, Michael F Plass, Nicholas H Briggs, and Rebecca L Braynard. 2009. Networking named content. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies*. ACM, 1–12.
[2] Xavier Marchal, Thibault Cholez, and Olivier Festor. 2016. Server-side performance evaluation of NDN. In *Proceedings of the 3rd ACM Conference on Information-Centric Networking*. ACM, 148–153.
[3] Ilya Moiseenko. 2014. Fetching content in Named Data Networking with embedded manifests. *NDN, Tech. Rep. NDN-0025* (2014).
[4] Stefa Lederer. 2015. Optimal Adaptive Streaming Formats MPEG-DASH HLS Segment Length. https://bitmovin.com/mpeg-dash-hls-segment-length [Online; accessed 22-August-2019].
[5] Wikipedia contributors. 2019. Merkle tree — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Merkle_tree&oldid=911089346 [Online; accessed 22-August-2019].