

Hash Chain: a Scalable Content Provenance and Integrity Verifying Protocol for NDN

ETRI

박세형 (labry@etri.re.kr)

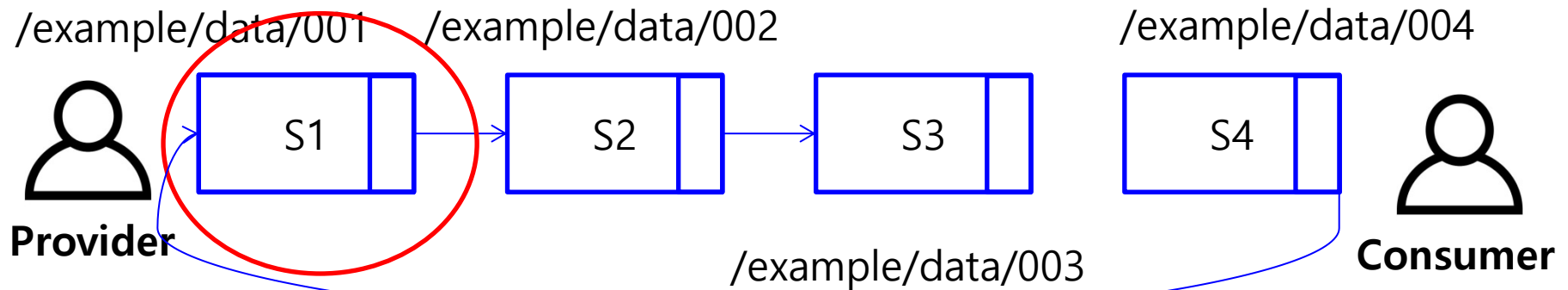
신용운 (uni2u@etri.re.kr)

June 29, 2020

Hash Chain light-weight per-packet authentication

- Hash Chain (HC) has two mechanisms.
 - **Batch Hash Chain**: This is the default mode that guarantee the provenance upon receiving the packet. However, the provider needs to have the entire sequences before generating hash chain signatures.
 - Real-time Hash Chain: This is apt for real-time usage. However, this cannot guarantee the provenance until it receives the last packet. For example, traffic signal of vehicular networks, other real-time tactile IoT signals

Backward Chain Signature Generation Example



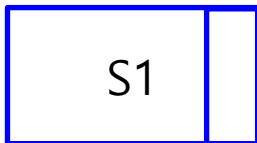
/example/data/004



$$K4 = H(\text{name4} \parallel \text{data4} \parallel 0x0000)$$

$$\text{Null_hash} = H(0x00000000000000000000000000000000)$$

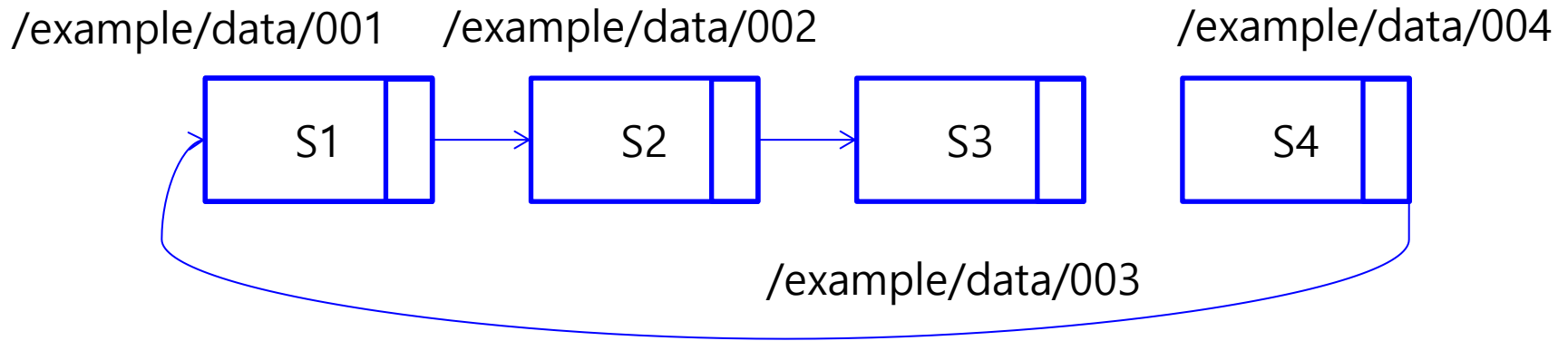
/example/data/001



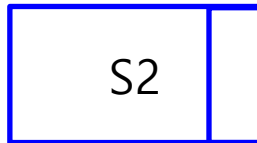
$$K1 = H(\text{name1} \parallel \text{data1} \parallel k4)$$

$$k4 = 0x66970e0d57360fdd4835c80$$

Backward Chain Signature Generation Example

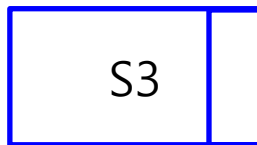


/example/data/002



$$K2 = H(\text{name2} \parallel \text{data2} \parallel k1)$$

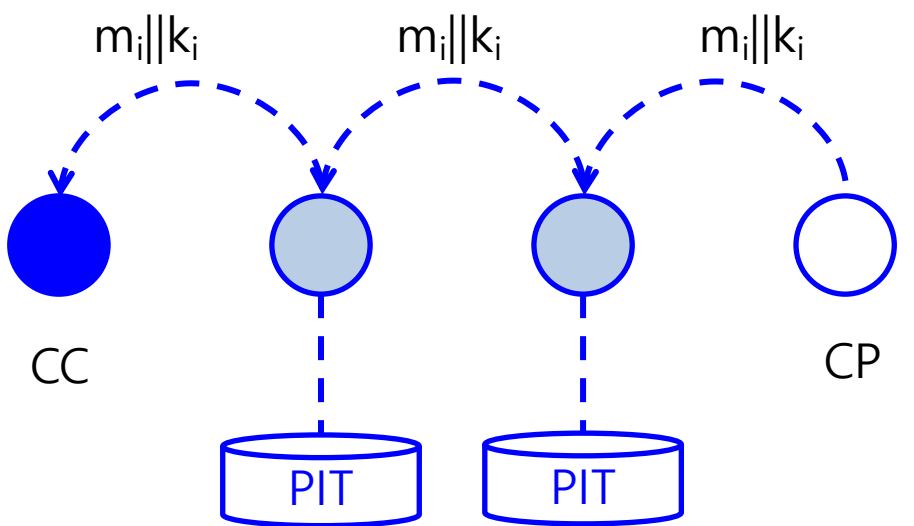
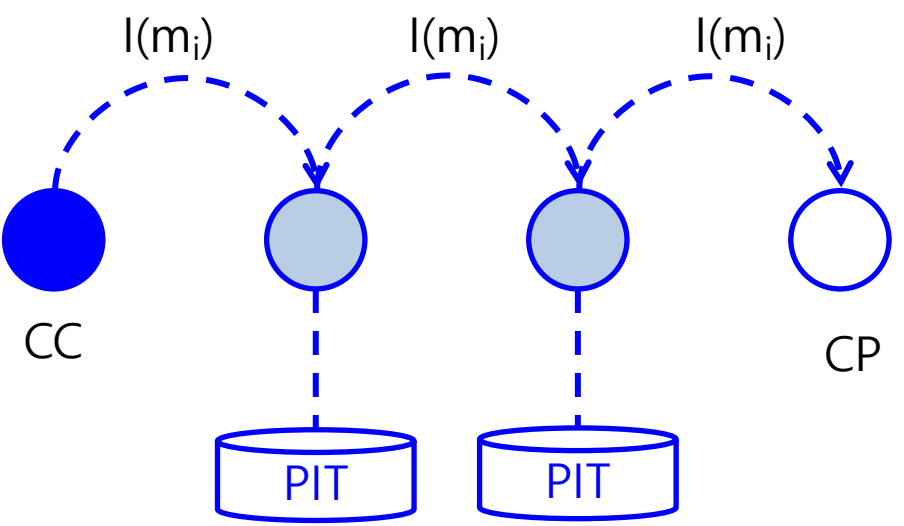
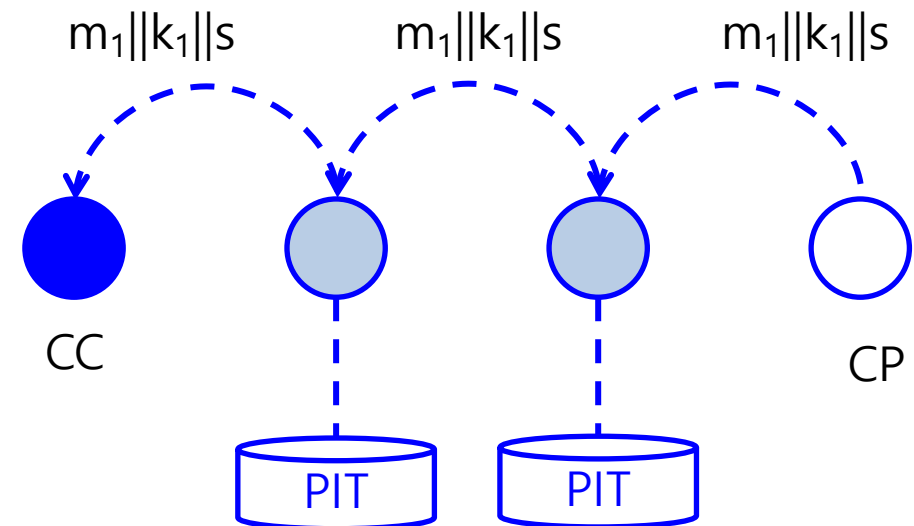
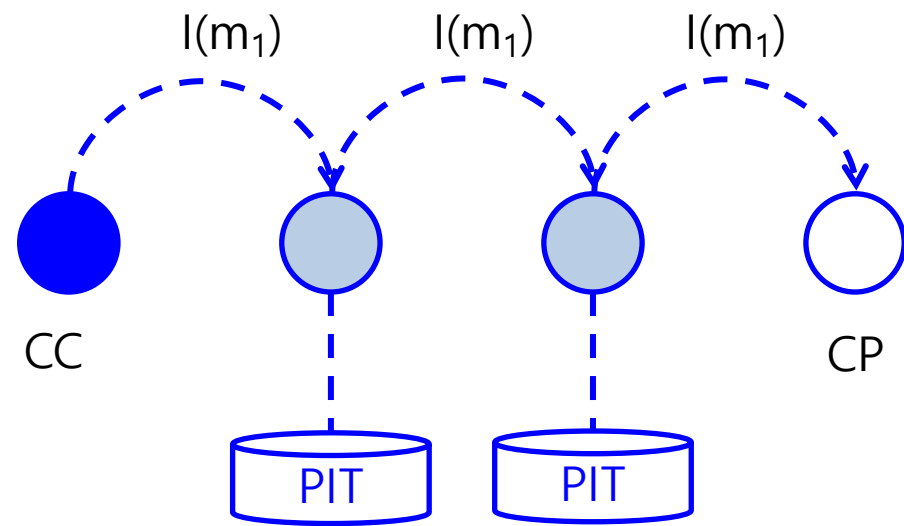
/example/data/003



$$K3 = H(\text{name3} \parallel \text{data3} \parallel k2)$$

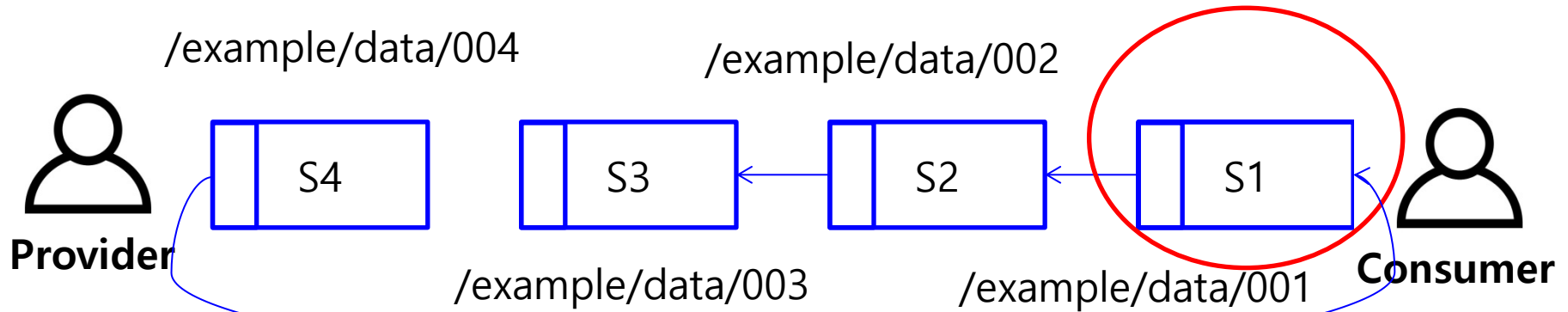
Backward Chain Signature Generation Example

- 0-step $m_1 = \text{name}_1 + \text{data}_1$
- (BC-1) $k_n = H(m_n \parallel \text{null_hash})$
- (BC-2) $k_1 = H(m_1 \parallel k_4)$
- (BC-3) $\text{RSA_SIGN}(\text{PR}_{\text{CP}}, K_1) \rightarrow$ maybe required to hash one more time
- (BC-4) $k_i = H(m_i \parallel k_{i-1})$, all $n > i > 1$
- (BC-5) $S = E(\text{PR}_{\text{CP}}, H(K_1) \parallel K_4)$

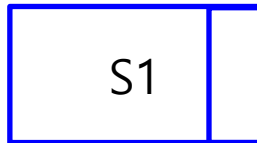


Propagation of Interest and Data packets for the Batch Hash Chain

Backward Chain Signature Verification Example



$/\text{example}/\text{data}/001$

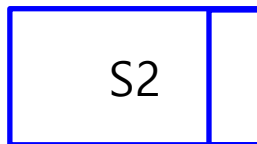


$$K4 = \text{Decrypt}(\text{PUB}_{\text{CP}}, S)$$

$$K1 = H(\text{name1} \parallel \text{data1} \parallel k4)$$

If $K1 == K1'$: pass

$/\text{example}/\text{data}/002$

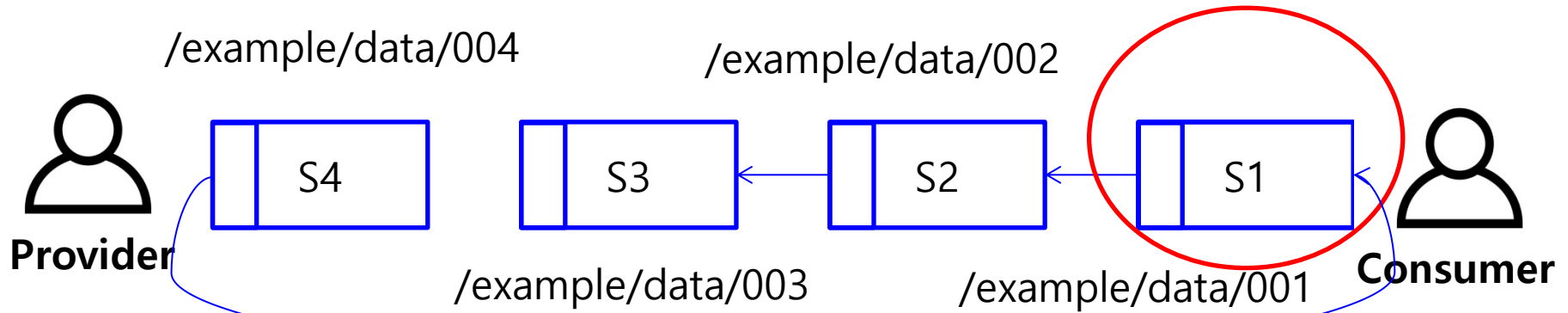


$$K2 = H(\text{name2} \parallel \text{data2} \parallel k1)$$

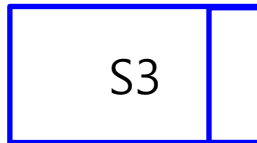
If $K2 == K2'$: pass

※ $K1'$ prime means calculated

Backward Chain Signature Verification Example



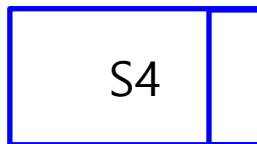
/example/data/003



$$K1 = H(\text{name3} \parallel \text{data3} \parallel k2)$$

If $K3 == K3'$: pass

/example/data/004



$$K4 = H(\text{name4} \parallel \text{data4} \parallel \text{null_hash})$$

If $K4 == K4'$: pass

Value	Reference	Description
0	DigestSha256	Integrity protection using SHA-256 digest
1	SignatureSha256WithRsa	Integrity and provenance protection using RSA signature over a SHA-256 digest
3	SignatureSha256WithEcdsa	Integrity and provenance protection using an ECDSA signature over a SHA-256 digest
4	SignatureHmacWithSha256	Integrity and provenance protection using SHA256 hash-based message authentication codes
5	SignatureSha256WithHashChain	Integrity and provenance protection using HashChain signature over a BLAKE3 digest
6	DigestBlake3	Integrity protection using Blake-3 digest
7	SignatureBlake3WithHashChain	Integrity and provenance protection using HashChain signature over a BLAKE3 digest
2,5-200		reserved for future assignments
>200		unassigned

BLAKE-3

- BLAKE-3 is compatible with SHA-256
 - <https://github.com/BLAKE3-team/BLAKE3>
 - <https://github.com/BLAKE3-team/BLAKE3/tree/master/c>
 - BLAKE3 is based on an optimized instance of the established hash function BLAKE2 and on the original Bao tree mode. The specifications and design rationale are available in the BLAKE3 paper. The default output size is 256 bits. The current version of Bao implements verified streaming with BLAKE3.

Implementation

- <https://named-data.net/doc/NDN-packet-spec/current/signature.html>
- Ndn-cxx HashChain
- [ndn-cxx/ndn-cxx/security/](#)
 - Along with signature-sha256-with-ecds
 - Digest-sha256
 - Digest-blake3
 - Signature-hash-chain-with-blake3
- Validation-policy and key-chain:
 - Make changes to validate hash chain
 - Make changes to validate Blake3

Conclusion

- Let's discuss!
- How to embed a Signature and keys
 - Signature in the data
 - (Data – size of signature)
 - Keys are placed as same as DigestSha256
 - https://named-data.net/doc/ndn-cxx/current/doxygen/d4/d08/sha256_8cpp_source.html