Anchored Secret Sharing for Access Control with Fast Revocation in Named Data Networking*

Jiangtao Luo[1,2], Chen He[2], Junxia Wang[2], Lianglang Deng, Yongyi Ran[1,2]

1. Electronic Information and Networking Research Institute, Chongqing University of Posts and Telecommunications, China

2. School of Communications and Information Engineering, Chongqing University of Posts and Telecommunications, China

Luojt@cqupt.edu.cn; 515308454@qq.com; 1552268578@qq.com; 13452005463@163.com; ranyy@cqupt.edu.cn

2021 4th International Conference on Hot Information-Centric Networking (HotICN) | 978-1-6654-0549-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/HOTICN53262.2021.9680834

***Abstract*—Named Data Networking (NDN), one typical representative of revolutionary future Internet architectures, enables users to obtain content from everywhere in the network, leveraging ubiquitous in-network caches, which facilitates content sharing more efficiently than the current Internet. However, it becomes a big challenge to implement access control in NDN since it is difficult for a content provider to know when, where and who has retrieved the content. The usual solution is to encrypt the entire content or each packet and then distribute the key to each subscribed user, but the cost of re-encryption to revoke the user is often significant. Inspired by the secret sharing method now commonly used in distributed storage for Big Data, we propose a novel scheme that can enforce access control to the entire content employing a piece of encoded data block, called *anchor share*. The scheme is thus named anchored secret share, abbreviated as anchor-SS. The performance of anchor-SS as well as its revocation efficiency is analyzed and evaluated. Testing results show that the re-encryption time will be significantly reduced in comparison to the popular approach, nearly inverse proportion to the number of shares.**

***Index Terms*—Information-Centric Networking, access control, revocation, secret sharing**

I. Introduction

While witnessing the unprecedented growth of users, traffic, and applications, the current TCP/IP-based Internet is also exposing its weakness in security, mobility, and energy efficiency, especially in scenarios of large-scale content sharing and massive connections. These concerns inspired researchers to redesign the Internet and as a result, a set of concerted achievements was worked out, termed as *Information-Centric Networking* (ICN), and gradually earned a great common consensus. In summary, ICN presents a clean-slate novel content-centric architecture rather than the traditional host-centric one, with remarkable features such as named data based addressing and routing, stateful forwarding, ubiquitous in-network caching, and object security [1].

Named Data Network (NDN) is the most popular implementation of ICN, which takes named data as the new "thin waist" [2] of the network layer and borrows the request-response communication model from HTTP. In NDN, consumers are able to conveniently retrieve content duplicates from the nearest cache mounted with routers besides the origin server. These features have brought convenience to some scenarios such as massive content sharing and a huge amount of connections. However, how to ensure that only legitimate users can access some confidential or sensitive content has become a new challenge.

Existing access control (AC) schemes in ICN/NDN can be broadly classified into two categories: *authentication-based* schemes and *encryption-based* ones. The former blocks requests from unauthorized users by means of authentication at network nodes. Qi Li et al. [3] proposed a lightweight integrity verification (LIVE) scheme that enables universal content signature verification and allows the CP to control content access in NDN nodes by selectively distributing integrity verification tokens to authorized nodes, so as to filter out those users without proper information at intermediate nodes. Later, another improved version was given [4], evolving the *token* into a so-called *capability* consisting of a signature and a token encoding access rights. Unfortunately, complex token management and extensive communications involved with the nodes undermined their scalability. Recently, authenticating at edge nodes is suggested to block unauthorized users at the very beginning, in which group signature is adopted to achieve anonymous authentication, appended with a revocation approach and a batch verification method [5], [6]. The biggest concern of authentication-based approaches is the overhead of authenticating every data request due to the lack of a connection, let alone vulnerabilities of nodes.

The latter, encryption-based solutions, are more popular [7], and their common feature is that the provider encrypts the content before distributing it to the network, and only legitimate users can get the corresponding key; what differs is the encryption method used. An early example is the dual-phase encryption (DPE) scheme [8], in which the edge service router re-encrypts the ciphered content with another key chosen randomly for each data request. The consumer has to derive the complete key (a function of those two encryption keys and the CP's private key) from the publisher. Kurihara et al. [9] provided another approach adopting object manifests to decouple encrypted content from access policy and key chains. Later, Li et al. [10] presented an attribute-based encryption

The work is jointly supported by the National Science Foundation of China (No. 62171072, 62172064, 62003067).

(ABE) access based control scheme, which can be divided into two levels: attribute management and an ABE-based naming. Recently, Misra et al. [11] proposed another access control framework, called *AccConF*, based on broadcast encryption (BE), in which the CP generates and distributes secret shares among legitimated users, and then disseminates an enabling block (EB) for users to extract keys using their own shares. The most impressive feature of AccConF is its achievements to implement user revocation by just updating the EB. The authors have presented a hybrid, fine-grained traceable and lightweight access control (TLAC) scheme [12], in which an anonymous and secure "three-way handshake" authentication protocol was presented by collaboratively leveraging the combined public key and the Schnorr signature [13], and an improved secret sharing method was used to distribute the key efficiently.

In most aforementioned methods, revocation cost was not trivial. In most cases, the CP has to re-encrypt the entire content and replace all duplicates cached all over the network for each revocation, which is an expensive task, especially for a large object. Only AccConF [11] is an exception with a relatively small cost to refresh the enabling block for each revocation. However, it has a limited number of revoked users, i.e., the CP has to re-key the whole system with a new polynomial when the number of revoked users reaches the polynomial's degree.

In one word, three major issues have to be addressed in scheming access control policy for NDN: 1) how to authorize only legitimated users to access the cached content and reject illegal users simultaneously; 2) how to protect the user's privacy as far as possible during the procedure; 3) how to revoke expired users efficiently.

In this paper, a novel hybrid scheme based on secret sharing of content is proposed, in which the content is split and encoded into several secret shares, then one of them is encrypted and taken as the anchor to control the access to the whole content.

Our contributions can be summarized as follows:

- We propose an AC scheme for NDN, adopting a fast (n, n) threshold secret sharing method by encrypting just one encoded chunk of the content (called *anchor share*). This scheme is the first solution combining XOR-code and secret shares as far as we know, which can cut down on the computation cost significantly in user revocation.
- For the sake of protecting user's privacy, we present an anonymous authentication mechanism through secure Interest packets during service subscription and cipher key distribution, utilizing *identity-based cryptography* (IBC) techniques.

The remainder of this paper is organized as follows. Section II outlines the related background technologies. Section III is devoted to the description of the proposed scheme. Section IV and V provide security analysis and performance evaluation, respectively. Finally, Section VI concludes this paper.

II. Preliminaries

A. NDN Basics

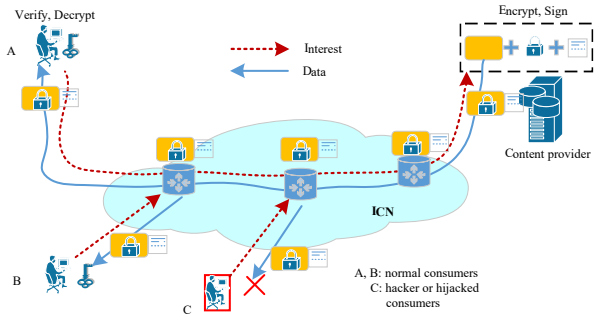


Fig. 1. Receiver-driven content retrieval, in-network caching, and content-based security in NDN.

In the NDN design, according to Fig. 1, every packet has a name by which it is forwarded other than the IP address of hosts. The communication in NDN is initiated by the receiving end, or data *consumer*, which sends out an Interest packet carrying a name that identifies the desired data [2]. When receiving an Interest packet, the intermediate router first checks its name in the local cache, called *Content Store* (CS). If the requested data has been buffered in the CS, a duplicate of Data packet will be sent back immediately to the consumer; otherwise, a new entry containing the name and the arrival interface will be appended to the *Pending Interest Table* (PIT); then the Interest packet is forwarded on through the output interface determined by the *Forwarding Information Base* (FIB), which is populated by a name-based routing protocol.

To sum up, the NDN design decouples information from its location. Instead of a socket-based communication model shaping the current Internet, NDN leverages the *request-response* model and in-network caches, enabling information retrieval to be fulfilled anywhere in the network. Such a change expedites content sharing, but in the meantime, poses great challenges to access control.

B. Secret Sharing

The *threshold secret sharing* scheme was first introduced by Shamir [14] and Blakley [15] in 1979 independently, which used to be utilized for sharing short keys among a set of participants, but now it has been commonly borrowed in distributed storage for Big Data. In Shamir's (k, n)-threshold method, a secret s is transformed into n shares, which can only be reconstructed from at least k $(\leq n)$ different shares; any grouping with less than k shares will not be able to reconstruct that content. The classical transformation method is through a randomly chosen $k-1$ degree polynomial: $q(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{k-1} x^{k-1}$, and all shares are represented by n points in the two-dimensional (2D) space (x_i, y_i), where $y_i = q(x_i)$, $i = 1, 2, \cdots, n$; a_0 represents the secret which can be reconstructed by classical Lagrange's Interpolation (Eqn. (1)) from at least k different points.

$$s = \sum_{i=1}^{k} y_i \times \prod_{j=1, j\neq i}^{k} \frac{-x_j}{x_i - x_j}, \tag{1}$$

As we know, Shamir's polynomial based solution is not viable for sharing a large amount of data due to its expensive communication and storage consumption, resulting from large share size equal to that of the whole secret (data), $O(n \log n)$ field operations in the phase of secret distribution, and $O(n^2)$ field operations in the phase of reconstruction [16]. Therefore, we select the Slepian-Wolf Coding (SWC) based secret share scheme [17] instead, leveraging lightweight exclusive-OR (XOR) network coding with reduced communication and storage costs.

C. Identity-Based Cryptography

IBC is a special case of *public key infrastructure* (PKI), in which any unique identity, such as a user's name, Email address, or their combinations, can be chosen as the public key, from which the private key can be calculated instantly by a *private key generator* (PKG) as requested after being successfully authenticated. Once public parameters are known, no further communications between clients and the server are required, differentiating IBC from other server-based cryptographic systems [18].

The IBC implementation is made up of two parts: *identity-based encryption* (IBE) and *identity-based signature* (IBS). In general, a hybrid IBE/IBS scheme is defined by six algorithms:

Setup: This algorithm configures the whole system by generating a pair of $(params, msk)$ from a security parameter, sp, by $(params, msk) = Setup(sp)$, where the public parameters $params$ will be publicly known while the master key msk is safely kept by the PKG only.

Extract: This algorithm is executed by the PKG to generate a secret key, K, as requested by a user with an identity of D using $K = \text{IBE}.extract(msk, params, D)$.

Encrypt: This algorithm is executed by the sender to cipher a plain message, M, with the recipient's identity of d, by $C = \text{IBE}.encrypt(M, d)$, where C is the generated ciphertext.

Sign: This algorithm is executed by the sender to sign the message with its private key, K, by $\sigma = \text{IBS}.sign(K, C)$, where σ represents the generated signature.

Verify: This algorithm is used by the recipient to confirm the message origin by $\text{IBS}.verify(D, \sigma, C)$, where D is the sender's identity, corresponding to the K.

Decrypt: The algorithm is used by the recipient to restore the original message by $M = \text{IBE}.decrypt(C, k)$, where k is its private key obtained from the PKG with its identity, d.

III. Proposed Solution

For the sake of simplicity, let's first define the coding operations.

Definition 1: XOR-Multiplication operation of Matrix. Given a matrix $\mathcal{X}$ with a dimension of $m \times m$, and a vector $\mathcal{Y}$ with a dimension of m, define the XOR-multiplication operation of $\mathcal{X}$ and $\mathcal{Y}$ is:

$$\mathcal{Z} = \mathcal{X} \circledast \mathcal{Y} = \begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,m} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,m} \\ \vdots & \vdots & \vdots & \vdots \\ x_{m,1} & x_{m,2} & \cdots & x_{2,m} \end{pmatrix} \circledast \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix}, \tag{2}$$

where, $\mathcal{X}$ is a binary matrix; each element of $\mathcal{Z}$, $z_i = (x_{i,1} \cdot y_1) \oplus (x_{i,2} \cdot y_2) \oplus \cdots \oplus (x_{i,m} \cdot y_m) = \oplus_{l=1}^{l=m} (x_{i,l} \cdot y_l)$, for each $i = 1, 2, \ldots, m$; $\oplus$ represents operation of bit-wise XOR.

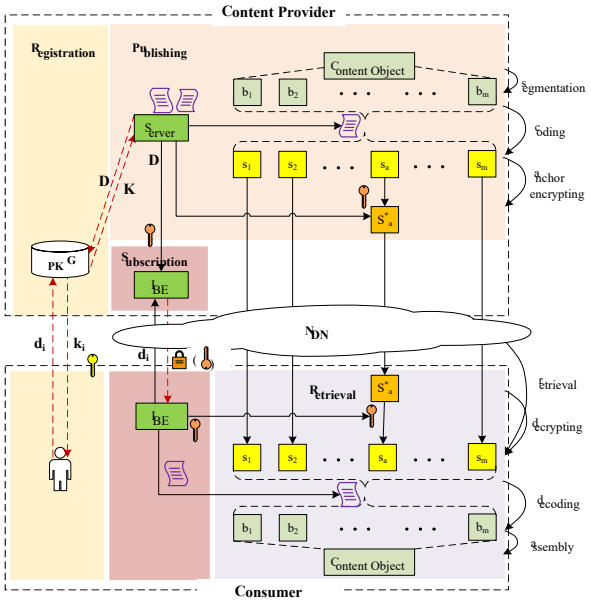


Fig. 2. Overview of the anchored secret sharing.

A. Threat Model and Assumptions

This paper focuses on the following two kinds of potential attacks.

- Attack by unregistered users. The unregistered users collect all data blocks and try to reconstruct the content.
- Attack by revoked users. These users try to construct the content with expired keys.

In the meantime, we must accept the following assumptions.

- Trustable CP and client applications. The CP and client programs will not kidnap the server or compromise the content and its keys.
- Semi-trustable routers and untrusted links. Intermediate routers are regarded reliable, while Links are high-risk.

B. Scheme of anchor-SS

According to Fig. 2, the scheme is roughly divided into four phases with reference to the publish/subscribe (Pub/Sub) framework: **registration**, **publishing**, **subscription**, and **retrieval**.

1) Registration: A new user is created with a new public identity d_i, an IBE key k_i, and some public parameters $params$ sent by the PKG through a secure channel, $k_i = \text{IBE}.extract(params, d_i)$, where i is the user index.

2) Publishing: Suppose the content length is L bits. It is first segmented into m blocks, denoted by b_j, each of which has a length of $|b_j| = L/m$ in bits, where $j = 1, 2, \ldots m$. If L cannot be divisible by m, pad '0' bit to b_1. The blocks are represented by a vector: $\mathcal{B} = \{b_1, b_2, \cdots, b_m\}^T$.

Next, the CP transforms the blocks by XOR-multiplication operation with a *coding matrix* (CM), obtaining the secret shares, $\mathcal{S} = \mathcal{C} \circledast \mathcal{B}$, where, $\mathcal{C}$ is the CM that will be explained in Section III-C.

Next, a certain share is randomly selected as the **anchor** for encryption, denoted by s_a. Considering the efficiency, symmetric encryption method (e.g., advanced encryption standard, AES) is utilized in our scheme to encrypt the anchor with a key. The key is called anchor key, denoted by ak, and the encrypted anchor $s_a^* = \text{AES}.encrypt(s_a, ak)$, where $\text{AES}.encrypt(\cdot)$ represents the encryption algorithm of AES. To expedite the decoding at the side of consumers, the CP will pre-compute for them the inverse of $\mathcal{C}$, called *decoding matrix*, $\mathcal{D} = \mathcal{C}^{-1}$.

Lastly, all shares including the encrypted anchor share are named after a certain naming rule and waiting for interested users to request.

3) Subscription: A user subscribes the content to the CP with its identity, denoted by d_i. The CP encrypts the anchor key with d_i as the IBE key, by $ak^* = \text{IBE}.encrypt(ak, d_i)$. The encrypted anchor key, along with the decoding matrix, $\mathcal{D}$, is sent to the user. The subscription has a certain expiration period (e.g., half-year), during which the anchor key is valid. When the subscription expires, the CP will re-encrypt the anchor with a new key and send the encrypted key to the renewing subscriber.

4) Retrieval: As usual, any consumer can collect all shares by sending Interest packets to the network. However, since the anchor share is encrypted, the content object cannot be reconstructed properly. An authorized consumer (e.g., u_i) can get the proper anchor key first by decrypting the encrypted one using its IBE private key by $ak = \text{IBE}.decrypt(ak^*, k_i)$, and then get the plaintext of anchor share by decrypting it with the obtained anchor key, $s_a = \text{AES}.decrypt(s_a^*, ak)$, where $\text{AES}.decrypt(\cdot)$ represents the decryption algorithm of AES.

Now, the consumer can further decode all plain shares with the decoding matrix $\mathcal{D}$ to get the original blocks, $\mathcal{B} = \mathcal{D} \circledast \mathcal{S}$, and reconstruct the content object.

C. Coding/Decoding Matrix

The coding matrix must fulfill the following requirements:

- Both $\mathcal{C}$ and $\mathcal{D}$ are binary matrices, i.e., all elements of them are 0 or 1.
- Determinants of $\mathcal{C}$ and $\mathcal{D}$ are equal to ± 1, i.e., $\det(\mathcal{C}) = \det(\mathcal{D}) = \pm 1$. This ensures that all rows or columns of $\mathcal{C}$ and $\mathcal{D}$ are linearly independent.
- The a-th column of $\mathcal{D}$ consists of all '1', which ensures all chunks are related to the anchor share.

Specifically, the coding matrix can be generated as follows. 1) Generate an m-order binary matrix Γ with a determinant of 1, i.e., $\det(\Gamma) = 1$; set elements of its a-th column all '1', where a is the index of the anchor share. In the simplest case, given an identity matrix I_m, set its a-th column all '1'. In practice, more '1's may be added to the identity $\mathcal{D}$, increasing difficulties to be hacked. 2) Calculate the inverse of $\mathcal{D}$ and get the coding matrix $\mathcal{C} = \mathcal{D}^{-1}$.

D. privacy protection

As described in III-B3, the user is required to submit its identity to the provider and may be confronted with privacy leaks. To address this, we design an anonymous authentication mechanism that uses the *secure Interest* to complete user service subscriptions.

1) Secure Interest: The so-called **secure Interest** packet is formed by adding two new header fields, `identity`, and `signature`, to a traditional Interest packet. Note that the field of `identity` contains not the consumer's real identity but an encrypted one.

Specifically, a consumer generates its encrypted identity by $d_i^* = \text{IBE}.encrypt(d_i, D)$, where, D is the public identity of the provider, and then puts it in the Interest packet and signs it with its private key, $\sigma = \text{IBS}.sign(k_i, sInt)$, forming the secure Interest $sInt$.

2) Anonymous Authentication: Upon receiving the secure Interest, the CP decrypts it using its IBE key to retrieve the requester's real identity (d_i), and verifies the signature to examine its validity using the retrieved identity. If the secure Interest is valid and the consumer is legitimate, other identity-specific procedures will be proceeded on, e.g., using the retrieved identity to encrypt the anchor key. Otherwise, the Interest will be discarded.

This mechanism can be used in any situation where an identity needs to be exchanged. In the subscription procedure, this mechanism is able to avoid transferring the plain-text of user identity through the network while authenticating.

IV. Analysis and Implementation Considerations

A. Security

Firstly, the security of XOR coding-based secret sharing scheme itself has been verified before. Specifically, in the (n, n)-threshold method, unauthorized users cannot reconstruct a content object without restoring even one of its shares. Therefore, it is viable to implement access control by just protecting one anchor share.

Secondly, the anchor share is AES ciphered while the cipher key is further IBE encrypted. Given a long enough key, it is extremely difficult to hack AES using brute force. In most cases, AES-128 is efficient and secure enough. Therefore, both the encrypted anchor share and its key can be cached at intermediate routers for later retrieval by legitimated consumers, since eavesdroppers can do nothing unless they could get proper IBE keys.

Finally, the PKG may be integrated with the content provider, avoiding key escrow, one of the biggest issues concerned for IBE.

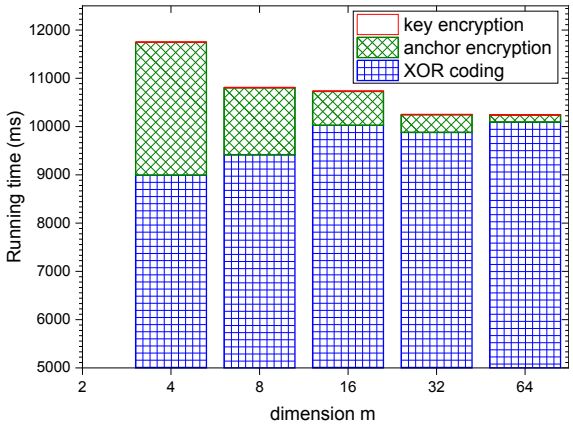


Fig. 3. Total runtime on matrix dimension: 1 GB file, $N = 2m - 1$.

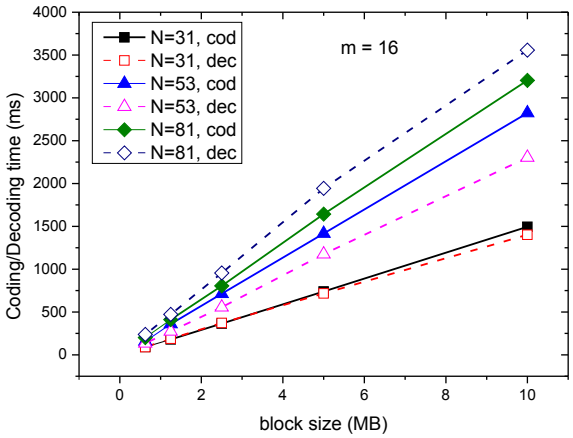


Fig. 4. Runtime vs. the block size and complexity of coding matrix.

B. Privacy

In service subscription, the user identity submitted to the network via a secure Interest packet is encrypted by IBE to ensure that the user's real identity cannot be exposed during the transfer and to keep eavesdroppers from connecting the identity with the subscribed content. Only the CP can obtain the real identity through IBE decryption after checking signatures; any requests with invalid signature or identity will be discarded. The vast majority of playback attacks can be filtered out by timestamps checking.

C. Revocation

When users' subscription expires, the CP will re-encrypt the anchor share with a new key, leaving the rest of encoded shares unchanged, without having to re-encrypt the entire content as in most other approaches. The old anchor shares cached in the network need to be cleared by means of intermediate nodes. This can be done automatically by setting the `freshness` field of the Data packet carrying those anchor shares.

On the other side, consumers can renew their subscriptions by requesting the new anchor key as well as the new version of encrypted anchor share.

V. Performance Evaluation

The efficiency and benefits of Anchor-SS were investigated in this section. Factors affecting its performance were explored first, and then in comparison to a popular approach, advantages of Anchor-SS were evaluated, in both content publishing and user revocation, as well as on both sides of CP and consumer. All tests were performed on a laptop equipped with 8 GB RAM, a 4-core CPU (frequency: 3.6 GHz) under Ubuntu Linux 18.04. All codes can be found on GitHub [19].

A. Performance Factors

This section examines the factors that affect performance, using the total time the CPU spends running all algorithms as a criterion.

At the CP side, the total runtime can be approximately described as $t_{\text{cp}} = t_{\text{cod}} + t_{\text{anc}} + t_{\text{key}}$, where t_{cod}, t_{anc}, and t_{key} denote the runtime for XOR-coding the blocks, encrypting the anchor share, and encrypting the anchor key, respectively. For a certain size of object file (e.g., 1 GB), they vary with the number of blocks, m, as shown in Fig. 3. t_{key} can generally be considered fixed; t_{anc} decreases proportionally to the multiplication of m; t_{key} keeps constant and nearly negligible. The total runtime tends to decrease and reach a stable value as m increases.

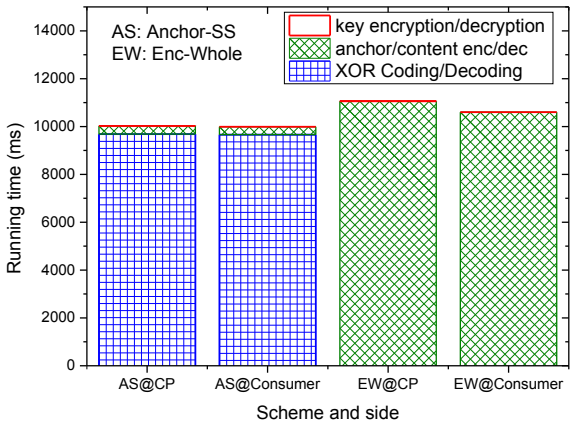
The XOR-coding and decoding time for different block size and complexities of coding matrices are illustrated in Fig. 4. The matrix complexity is described by the number of non-zero elements (i.e. '1') in the matrix, denoted by N. It is observed that both the coding and decoding time rise linearly as the block size increases for all Ns. And a larger N brings longer coding and decoding time; thus, for a certain m, the least coding time is obtained when selecting the simplest coding matrix, i.e., $N = 2m - 1$.

B. Performance Benefits

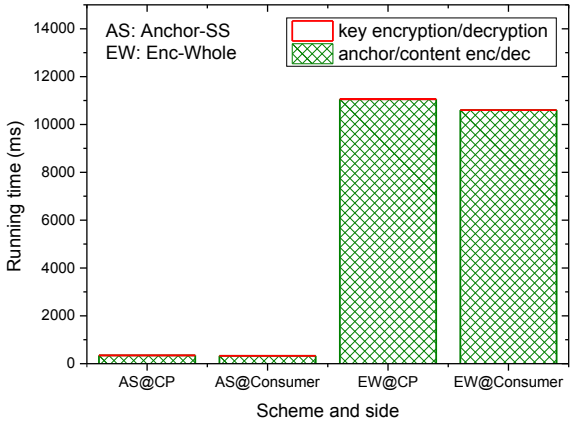
In this section, the benefits of our proposed anchor-SS are investigated. A popular approach to secure content delivery is to encrypt the whole file and then transfer the key to users, denoted by *Enc-Whole*, which is taken as a baseline for performance evaluation. For the sake of fairness, same encryption methods are utilized: AES-128 for the anchor share or the content, and IBE for the key. Given the test file size is 1 GB, which is split into $m = 32$ blocks.

The runtime using two schemes are illustrated and compared in Fig. 5. At the side of consumer, similarly, the total runtime can be approximately described by $t_{\text{cc}} = t'_{\text{key}} + t'_{\text{anc}} + t_{\text{dec}}$, where t'_{key} and t'_{anc} are respectively for the runtime of decrypting anchor key and anchor share, and t_{dec} for XOR-decoding the shares.

According to Fig. 5a, in the case of content publishing and retrieval, the total runtime of Anchor-SS is sightly less than that of Enc-Whole. In terms of composition, in anchor-SS, the XOR codec consumes the vast majority of the runtime, while in Enc-Whole, the content encryption and decryption do the same thing. In particular, the time consumed to encrypt the



(a) Publishing & retrieval



(b) Re-encryption for user revocation

Fig. 5. Computation cost comparison of Anchor-SS and Enc-Whole ($m = 32, N = 63$).

anchor share is $1/m$ of encrypting the whole file, which is equivalent to the ratio of the anchor share to the whole file size. It is strengthened again in the procedure of revocation, as shown in Fig. 5b. Compared to Enc-Whole, the anchor-SS requires only about $1/m$ of runtime, due to the elimination of XOR codec in revocation.

VI. Conclusion

In this work, a threshold based secret shares scheme was proposed for access control in ICN. The proposed scheme was verified to ensure secure content delivery by encrypting a randomly selected share, called *anchor share*, and re-encrypting it for user revocation to earn grossly m-fold increase in the revocation efficiency, compared to the popular approach, with m representing the number of shares.

In future research, cluster-based anchor-SS or generation of a new anchor may be further explored to avoid invalidating all cached anchor shares throughout the network. Additionally, the proposed scheme may be expanded to content distribution scenarios other than ICN.

References

[1] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, "A survey of information-centric networking," *Communications Magazine, IEEE*, vol. 50, no. 7, pp. 26–36, July 2012.

[2] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, k. claffy, P. Crowley, C. Papadopoulos, L. Wang, and B. Zhang, "Named data networking," *SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 3, pp. 66–73, Jul. 2014. [Online]. Available: http://doi.acm.org/10.1145/2656877.2656887

[3] Q. Li, X. Zhang, Q. Zheng, R. Sandhu, and X. Fu, "LIVE: Lightweight integrity verification and content access control for named data networking," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 2, pp. 308–320, Feb 2015.

[4] Q. Li, P. P. C. Lee, P. Zhang, P. Su, L. He, and K. Ren, "Capability-based security enforcement in named data networking," *IEEE/ACM Transactions on Networking*, vol. 25, no. 5, pp. 2719–2730, Oct 2017.

[5] K. Xue, X. Zhang, Q. Xia, D. S. L. Wei, H. Yue, and F. Wu, "SEAF: A secure, efficient and accountable access control framework for information centric networking," in *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, April 2018, pp. 2213–2221.

[6] K. Xue, P. He, X. Zhang, Q. Xia, D. S. L. Wei, H. Yue, and F. Wu, "A secure, efficient, and accountable edge-based access control framework for information centric networks," *IEEE/ACM Transactions on Networking*, vol. 27, no. 3, pp. 1220–1233, June 2019.

[7] R. Tourani, S. Misra, T. Mick, and G. Panwar, "Security, privacy, and access control in information-centric networking: A survey," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 1, pp. 566–600, Firstquarter 2018.

[8] Q. Zheng, G. Wang, R. Ravindran, and A. Azgin, "Achieving secure and scalable data access control in information-centric networking," in *Communications (ICC), 2015 IEEE International Conference on*, June 2015, pp. 5367–5373.

[9] J. Kuriharay, E. Uzun, and C. A. Wood, "An encryption-based access control framework for content-centric networking," in *IFIP Networking Conference (IFIP Networking), 2015*, May 2015, pp. 1–9.

[10] B. Li, D. Huang, Z. Wang, and Y. Zhu, "Attribute-based access control for icn naming scheme," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 2, pp. 194–206, March 2018.

[11] S. Misra, R. Tourani, F. Natividad, T. Mick, N. E. Majd, and H. Huang, "AccConF: An access control framework for leveraging in-network cached data in the ICN-enabled wireless edge," *IEEE Transactions on Dependable and Secure Computing*, vol. 16, no. 1, pp. 5–17, 2019.

[12] J. Luo, C. He, and J. Wang, "Traceable lightweight and fine-grained access controlin named data networking (in Chinese)," vol. 41, no. 10, pp. 2428–2434.

[13] C. P. Schnorr, "Efficient signature generation by smart cards," *Journal of Cryptology*, vol. 4, no. 3, pp. 161–174, 1991.

[14] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979. [Online]. Available: http://doi.acm.org/10.1145/359168.359176

[15] G. R. Blakley, "Safeguarding cryptographic keys," in *Managing Requirements Knowledge, International Workshop on*. Los Alamitos, CA, USA: IEEE Computer Society, 6 1979, p. 313. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/AFIPS.1979.98

[16] Y. Wang, "Privacy-preserving data storage in cloud using array bp-xor codes," *IEEE Transactions on Cloud Computing*, vol. 3, no. 4, pp. 425–435, Oct 2015.

[17] T. Tran, M. Rahman, M. Z. A. Bhuiyan, A. Kubota, S. Kiyomoto, and K. Omote, "Optimizing share size in efficient and robust secret sharing scheme for big data," *IEEE Transactions on Big Data*, pp. 1–1, 2017.

[18] G. Appenzeller, "Identity-based encryption architecture and supporting data structures," IETF RFC 5408, January 2009.

[19] Anchor-ss. [Online]. Available: https://github.com/jiangtaoluo/anchor