論文 / 著書情報
Article / Book Information

Title	Chosen-ciphertext secure code-based threshold public key encryptions with short ciphertext
Authors	Kota Takahashi, Keitaro Hashimoto, Wakaha Ogata
Citation	Designs, Codes and Cryptography
Pub. date	2023, 10
Creative Commons	Information is in the article.

Designs, Codes and Cryptography
https://doi.org/10.1007/s10623-023-01309-2

Chosen-ciphertext secure code-based threshold public key encryptions with short ciphertext

Kota Takahashi[1] · **Keitaro Hashimoto**[2] · **Wakaha Ogata**[1]

Received: 27 March 2023 / Revised: 27 March 2023 / Accepted: 13 September 2023

Abstract
Threshold public-key encryption (threshold PKE) has various useful applications. A lot of threshold PKE schemes are proposed based on RSA, Diffie–Hellman and lattice, but to the best of our knowledge, *code-based* threshold PKEs have not been proposed. In this paper, we provide three IND-CCA secure code-based threshold PKE schemes. The first scheme is the concrete instantiation of Dodis–Katz conversion (Dodis and Katz, TCC'05) that converts an IND-CCA secure PKE into an IND-CCA secure threshold PKE using parallel encryption and a signature scheme. This approach provides non-interactive threshold decryption, but ciphertexts are large (about 16 kilobytes for 128-bit security) due to long code-based signatures even in the state-of-the-art one. The second scheme is a new parallel encryption-based construction without signature schemes. Unlike the Dodis–Katz conversion, our parallel encryption converts an OW-CPA secure PKE into an OW-CPA secure threshold PKE. To enhance security, we use Cong et al.'s conversion (Cong et al., ASIACRYPT'21). Thanks to eliminating signatures, its ciphertext is 512 bytes, which is only 3% of the first scheme. The decryption process needs an MPC for computing hash functions, but decryption of OW-CPA secure PKE can be done locally. The third scheme is an MPC-based threshold PKE scheme from code-based assumption. We take the same approach Cong et al. took to construct efficient lattice-based threshold PKEs. We build an MPC for the decryption algorithm of OW-CPA secure Classic McEliece PKE. This scheme has the shortest ciphertext among the three schemes at just 192 bytes. Compared to the regular CCA secure Classic McEliece PKE, the additional ciphertext length is only 100 bytes. The cons are heavy distributed computation in the decryption process.

Communicated by E. Gorla.

✉ Wakaha Ogata
ogata.w.aa@m.titech.ac.jp

Kota Takahashi
takahashi.kota.kt@gmail.com

Keitaro Hashimoto
keitaro.hashimoto@aist.go.jp

[1] Tokyo Institute of Technology, Tokyo, Japan

[2] National Institute of Advanced Industrial Science and Technology (AIST), Tokyo, Japan

Published online: 12 October 2023

Keywords Threshold public-key encryption · Threshold decryption · Parallel encryption · Code-based cryptography · Classic McEliece

Mathematics Subject Classification 94A60 · 68P25

1 Introduction

1.1 Background

Threshold public-key encryption is a variant of public-key encryption (PKE) where the decryption key is distributed to multiple parties as partial decryption keys. A ciphertext is decrypted by a threshold decryption protocol among parties. If more than a certain threshold number of, but not necessarily all, parties execute the protocol, the ciphertext is correctly decrypted. On the other hand, parties below the threshold cannot obtain any information about the plaintext from a ciphertext even if they have their own partial decryption key.

We enjoy the features of threshold PKEs in various applications, for example, e-voting, blockchain, and threshold implementation. In many e-voting systems such as [1, 15, 18, 32], each ballot is encrypted by a threshold PKE. So, voters' privacy is protected from (a few) curious tally servers, and simultaneously, the voting results can be obtained by all (or most) tally servers jointly decrypting non-private information.

Also, some blockchain applications [33, 48] use threshold PKEs. For example, in [48], to prevent a miner from abusing knowledge from transactions before being added to the chain, encrypted transactions are added to the chain first, and then randomly selected miners jointly decrypt them. The property of threshold PKEs ensures that no one knows the content of transactions before they are added, and also that no one cannot interfere with the publishing of inconvenient transactions. Another interesting application of threshold PKEs is a countermeasure against side-channel attacks, known as "Threshold Implementation" [39]. By implementing the threshold decryption algorithm in a single device (i.e., virtual multiple decryptors run the decryption process in it), key recovery using side-channel information (e.g., power spectrum) becomes significantly hard [10, Section 2.5].

Because of such use of threshold PKEs, the US National Institute of Standards and Technology (NIST) is planning standardization of threshold PKEs [37] in order to encourage implementations of threshold PKEs in the real world.

In the literature, a lot of threshold PKE schemes have been proposed from various computational assumptions, for example, RSA [28], composite residuosity [21], discrete logarithm [23] and Diffie–Hellman [12, 43]. However, these schemes are not secure against quantum computers due to Shor's algorithm [42], which enables quantum computers to solve factoring and discrete logarithm in polynomial time. To ensure security for the future, it is important to construct *post-quantum* threshold PKE schemes, which are based on intractable problems even against quantum computers.

Some post-quantum threshold PKE schemes were proposed in the area of lattice-based cryptography [6, 16, 34]. On the other hand, there are no known post-quantum threshold PKEs from other assumptions, e.g., code, multivariate, and isogenies. In case lattice-based assumptions no longer hold, it is desirable to construct post-quantum threshold PKEs from different assumptions as well. In fact, due to the same reason, NIST is still in the process of selecting code-based KEMs in the fourth round of the PQC competition after NIST selected

a lattice-based KEM [38]. Therefore, constructing post-quantum threshold PKEs other than lattice-based ones is an important issue.

1.2 Prior work

A naive approach to constructing a post-quantum threshold PKE is computing the decryption algorithm of a non-threshold PKE by multi-party computations (MPCs). Theoretically speaking, for any decryption algorithm, its threshold decryption can be realized by MPCs; but the obtained protocol is *impractical* in general. Consider the case of hybrid encryption (i.e., KEM-DEM framework) often used to efficiently encrypt long messages. Hybrid encryption leverages the high performance of DEM's encryption and decryption process to achieve good performance in total. However, replacing the entire decrypting process with an MPC requires DEM decryption to be performed in a threshold fashion, which takes huge costs, especially for long messages, even if the MPC for the KEM is lightweight.

To realize *practical* post-quantum threshold PKEs, we can use two known generic conversions, Dodis–Katz conversion [24] and Cong et al. conversion (CCMS conversion) [16].

Dodis–Katz conversion transforms an IND-CCA secure PKE to an IND-CCA secure threshold PKE using the concept of parallel encryption. Parallel encryption is a scheme that divides a plaintext into n pieces and encrypts each piece with a distinct public key of the underlying PKE. In Dodis–Katz threshold PKEs, the ciphertext consists of n ciphertext and a one-time signature with strong EUF-CMA security, which is required to achieve CCA security. To decrypt a ciphertext, each party first recovers the corresponding piece and then recovers the message from the collection of the pieces. Since the decryption process only does local computation (i.e., without MPCs), resulting threshold PKEs are optimal in the sense of computation cost. Also, since the conversion can start from any regular IND-CCA secure PKE, we can obtain post-quantum threshold PKEs from various post-quantum assumptions.

Recently, Cong et al. presented an elegant KEM-DEM framework (called CCMS conversion in this paper) that can be used to efficiently convert an OW-CPA secure *deterministic* threshold PKE into an IND-CCA secure threshold PKE. CCMS conversion can be considered as a variant of the Fujisaki–Okamoto transformation (FO transformation) proposed by [29] that converts an OW-CPA secure *probabilistic* PKE into an IND-CCA one. More precisely, a deterministic encryption function $\mathsf{Enc}_{\mathsf{p}}$ is first converted into a probabilistic one as $\mathsf{Enc}'_{\mathsf{p}}(\mathsf{pk}, \mathsf{m}; r) := (\mathsf{Enc}_{\mathsf{p}}(\mathsf{pk}, \mathsf{m}), r)$, and then applied FO transformation.[1] In the decryption process of the resulting PKE, the ciphertext validity can be verified simply by checking hash values before decrypting the DEM part. Moreover, it has an attractive property that, if the validity checks are passed, it is still secure even if the session key of DEM becomes public (i.e., all parties know it). From this property, it is sufficient only to compute the decryption of KEM in a distributed fashion, and the decryption of the DEM can be done in the clear. Therefore, once we construct an OW-CPA secure (deterministic) PKE equipping an efficient distributed decryption, we can obtain an IND-CCA threshold PKE, which is much simpler than constructing an MPC for a more complex decryption algorithm of IND-CCA secure PKEs.

[1] In order to guarantee the security under threshold setting, small modification has been introduced to design CCMS conversion.

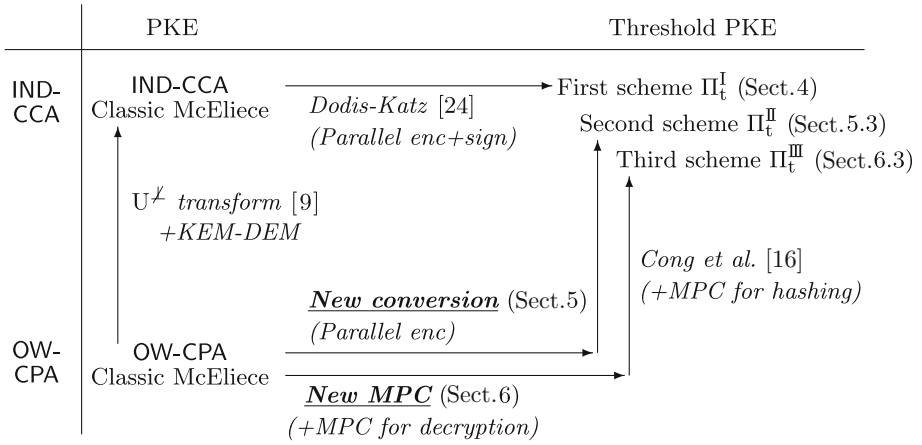

Fig. 1 The Classic McEliece-based threshold PKE schemes Π_t^{I}, Π_t^{II} and Π_t^{III}. The arrows ($\longrightarrow$) indicate the corresponding conversions. Our proposed conversions are highlighted with underlining

1.3 Our contributions

In this work, to ensure the diversity of post-quantum threshold PKEs, we provide three IND-CCA secure *code-based* threshold PKE schemes, Π_t^{I}, Π_t^{II}, and Π_t^{III}. These schemes are all based on Classic McEliece [2], one of the candidates in the NIST PQC standardization. Figure 1 shows how they are constructed from (OW-CPA secure) Classic McEliece PKE.

The first scheme Π_t^{I} is the concrete instantiation of Dodis–Katz conversion from code-based assumptions. We instantiate it with the IND-CCA secure Classic McEliece PKE[2] and the strong EUF-CMA secure Sig 3 signature [8], which is the state-of-the-art signature scheme based on the same post-quantum assumption as Classic McEliece. We reveal that Π_t^{I} has a large ciphertext (about 16 kilobytes for 128-bit security[3]) since the size of code-based signatures is large.

The second scheme Π_t^{II} is a new parallel encryption-based construction without signature schemes. To obtain Π_t^{II}, we start from OW-CPA Classic McEliece PKE, convert it into a threshold one (with OW-CPA security) using newly-developed parallel encryption, and then enhance its security into IND-CCA using CCMS conversion. Since CCMS requires a deterministic PKE, the new parallel encryption must preserve the deterministic property of the underlying PKE, while it does not need to guarantee CCA security. Our idea to realize such a conversion is; (1) utilizing a simple dividing method instead of a threshold secret sharing scheme used in Dodis–Katz parallel encryption, (2) assigning multiple key pairs to each decryption party to support t out of n setting, and (3) eliminating unnecessary signatures. Thanks to eliminating signatures, Π_t^{II} drastically reduces the ciphertext length: it is 512 bytes, which is only 3% of the ciphertext length of Π_t^{I}. Although Π_t^{II} provides smaller ciphertext, Π_t^{II} needs an MPC for computing hash functions during the decryption process. However, it does not need MPCs for the decryption of the OW-CPA secure PKE since we use parallel encryption techniques.

[2] IND-CCA secure Classic McEliece PKE is constructed from IND-CCA secure Classic McEliece KEM and IND-CCA secure SKE via the KEM-DEM framework.

[3] Throughout this paper, concrete values are given in 128-bit security.

The third scheme Π_t^{III} is an MPC-based threshold PKE scheme from code-based assumption. We take the same approach Cong et al. took to construct efficient lattice-based threshold PKEs; prepare OW-CPA secure threshold PKEs by building MPCs for key generation and decryption, and convert them into CCA ones. We build an MPC for computing the decryption algorithm of OW-CPA secure Classic McEliece PKE. Most part of the decryption algorithm is a decoding process of the Goppa code, which is done by Patterson's algorithm. We notice that it is comparatively MPC-friendly because of its algebraic computation. In contrast to other decoding algorithms for e.g., MDPC codes and LRPC codes, Patterson's decoding algorithm does not use integer comparison or operations on basis vectors, which require heavy MPCs. Almost all parts of Patterson's decoding can be computed in a distributed fashion by existing MPC techniques, but computing the extended Euclidean Algorithm part is non-trivial. We succeed in constructing an MPC for it using the idea of exploiting the properties of the Subresultant Matrix used in [36]. As a result, we obtain an OW-CPA secure threshold Classic McEliece PKE from our new MPCs for the decryption. Then, we convert it into CCA secure one Π_t^{III} via CCMS conversion. Π_t^{III} has the shortest ciphertext length among the three schemes at just 192 bytes. Compared to the regular CCA secure Classic McEliece PKE, the additional ciphertext length is only 100 bytes. The cons of Π_t^{III} is heavy distributed computation in the decryption process. It invokes a lot of interaction to compute Patterson's algorithm via MPCs.

1.4 Organization of this paper

This paper is organized as follows. In Sect. 2, we introduce notations and the syntax and the security notions of cryptographic primitives used in this paper. In Sect. 3, we explain the background of code-based cryptography. We explain Classic McEliece PKE/KEM along with the Goppa code and Patterson decoding, and related works about code-based signatures. Then, in the subsequent three sections, we explain the concrete code-based threshold PKE schemes. Section 4 describes the concrete instantiation of Dodis–Katz conversion from code-based assumption. Section 5 explains a new parallel encryption-based threshold PKE. Section 6 explains a new MPC-based threshold PKE from Classic McEliece, including the concrete procedure to compute Patterson's decoding algorithm in distributed fashions. Finally, in Sect. 7, we compare the three schemes in terms of ciphertext length and computation complexity of threshold decryption. Conclusions are provided in Sect. 8.

2 Preliminaries

2.1 Notations

Let λ be a security parameter. For a non-negative integer n, $[n]$ denotes $\{1, 2, \ldots, n\}$. $\binom{n}{t}$ denotes the number of t-combinations for n elements, i.e., $\binom{n}{t} := \frac{n(n-1)\cdots(n-t+1)}{t(t-1)\cdots 1}$. For finite set X, $x \leftarrow_{\$} X$ indicates an element $x \in X$ is chosen uniformly at random. QPT stands for quantum polynomial time.

2.2 Public-key encryption

A public key encryption (PKE) scheme is defined as a triple of algorithms $\Pi_{\mathrm{p}} = (\mathsf{KGen}_{\mathrm{p}}, \mathsf{Enc}_{\mathrm{p}}, \mathsf{Dec}_{\mathrm{p}})$ with plaintext space M_{p} and ciphertext space C_{p}. The key generation algorithm $\mathsf{KGen}_{\mathrm{p}}$ takes a security parameter 1^{λ} as input and outputs a pair of public key and secret key $(\mathsf{pk}, \mathsf{sk})$. The encryption algorithm $\mathsf{Enc}_{\mathrm{p}}(\mathsf{pk}, \mathsf{m})$ computes a ciphertext ct from a plaintext $\mathsf{m} \in M_{\mathrm{p}}$ and the public key pk. Note that we treat a deterministic PKE throughout this paper and thus we suppose that $\mathsf{Enc}_{\mathrm{p}}$ is a deterministic algorithm. The decryption algorithm $\mathsf{Dec}_{\mathrm{p}}(\mathsf{sk}, \mathsf{ct})$ recovers a plaintext $\mathsf{m} \in M_{\mathrm{p}}$ or returns the special symbol $\perp \notin M_{\mathrm{p}}$.

A probability of decryption failure (for a randomly chosen plaintext) δ_f is defined as

$$\delta_f := \Pr\left[\mathsf{Dec}_{\mathrm{p}}(\mathsf{sk}, \mathsf{Enc}_{\mathrm{p}}(\mathsf{pk}, \mathsf{m})) \neq \mathsf{m}\right],$$

where $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_{\mathrm{p}}(1^{\lambda})$ and $\mathsf{m} \leftarrow_{\$} M_{\mathrm{p}}$. We say that Π_{p} is $(1-\delta_f)$-correct. We require that δ_f is exponentially small. If $\delta_f = 0$, we say Π_{p} is perfectly correct. When Π_{p} fails to decrypt, there are two types of decryption failures. The first type of decryption failure is two different plaintexts are encrypted to the same ciphertext. We say that Π_{p} is δ_c-*collision free* when

$$\Pr\left[\exists \mathsf{m}_1, \mathsf{m}_2 \in M_{\mathrm{p}} \text{ s.t. } \mathsf{Enc}_{\mathrm{p}}(\mathsf{pk}, \mathsf{m}_1) = \mathsf{Enc}_{\mathrm{p}}(\mathsf{pk}, \mathsf{m}_2)\right] = \delta_c,$$

where $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_{\mathrm{p}}(1^{\lambda})$. If Π_{p} is perfectly correct, then it is 0-collision free. The second type of decryption failure occurs when a valid ciphertext is decrypted to $\perp$. We define a game called $\perp$-aware in which an adversary $\mathcal{A}$ given the public key pk finds a pair of plaintext and ciphertext $(\tilde{\mathsf{m}}, \tilde{\mathsf{ct}})$ such that $\tilde{\mathsf{ct}} = \mathsf{Enc}_{\mathrm{p}}(\mathsf{pk}, \tilde{\mathsf{m}})$ but $\mathsf{Dec}_{\mathrm{p}}(\mathsf{sk}, \tilde{\mathsf{ct}}) = \perp$. The advantage of $\mathcal{A}$ is defined as

$$\mathsf{Adv}^{\perp\text{-aware}}_{\Pi_{\mathrm{p}}, \mathcal{A}} := \Pr\left[\tilde{\mathsf{ct}} = \mathsf{Enc}_{\mathrm{p}}(\mathsf{pk}, \tilde{\mathsf{m}}) \wedge \mathsf{Dec}_{\mathrm{p}}(\mathsf{sk}, \tilde{\mathsf{ct}}) = \perp\right],$$

where $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_{\mathrm{p}}(1^{\lambda})$. We say that Π_{p} is $\delta_{\perp}$-$\perp$-aware, if the above advantage is $\delta_{\perp}$ or less for any QTP adversary $\mathcal{A}$. If Π_{p} is perfectly correct, it is 0-$\perp$-aware.

Moreover, we say that Π_{p} is *rigid* if it satisfies the following condition [7]: For any $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_{\mathrm{p}}(1^{\lambda})$ and $\mathsf{ct} \in C_{\mathrm{p}} \backslash C_{\mathrm{p}}^{\perp}$,

$$\Pr\left[\mathsf{Enc}_{\mathrm{p}}(\mathsf{pk}, \mathsf{Dec}_{\mathrm{p}}(\mathsf{sk}, \mathsf{ct})) = \mathsf{ct}\right] = 1$$

holds, where $C_{\mathrm{p}}^{\perp} \subset (C_{\mathrm{p}})$ is the set of all ciphertexts $\mathsf{ct} \in C_{\mathrm{p}}$ for which $\mathsf{Dec}_{\mathrm{p}}(\mathsf{sk}, \mathsf{ct}) = \perp$.

The standard security notion of public key encryption is indistinguishability (IND) against chosen ciphertext attack (CCA). This security guarantees that an adversary cannot obtain any information about its plaintext from a ciphertext even if it accesses the decryption oracle. Let $\mathcal{A}$ be an adversary. Given a public key pk, $\mathcal{A}$ accesses to the decryption oracle O_{Dec}, and outputs two plaintexts $\mathsf{m}_0^*, \mathsf{m}_1^*$. Then one of the two plaintexts is encrypted to the challenge ciphertext $\mathsf{ct}^* = \mathsf{Enc}_{\mathrm{p}}(\mathsf{pk}, \mathsf{m}_b^*)$ based on a randomly chosen challenge bit b. $\mathcal{A}$ is given the challenge ciphertext ct^* and the access to O_{Dec}, and it outputs a bit b'. If the advantage of $\mathcal{A}$, given by

$$\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\Pi_{\mathrm{p}}, \mathcal{A}} := \left| 2 \cdot \Pr\left[b = b'\right] - 1 \right|$$

is negligible for all QPT $\mathcal{A}$, we say that Π_{p} is IND-CCA secure.

To design an IND-CCA PKE scheme, we sometimes use a PKE scheme with weaker security, called onewayness (OW) against chosen plaintext attacks (CPA). OW-CPA security ensures that an adversary cannot recover the whole plaintext from a ciphertext. This adversary

$\mathcal{A}$ is given a challenge ciphertext ct^* whose plaintext m^* is chosen randomly from the plaintext space M_p. Then $\mathcal{A}$ guesses m^*. The advantage of $\mathcal{A}$ is defined as follows:

$$\mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\Pi_\mathrm{p},\mathcal{A}} := \left| \Pr\left[\mathsf{m} = \mathsf{m}^*\right] - \frac{1}{|M_\mathrm{p}|} \right|$$

If the advantage is negligible for all QPT $\mathcal{A}$, we say that Π_p is OW-CPA secure.

2.3 Threshold public-key encryption

A (t, n)-*threshold* public-key encryption scheme is an extension of public-key encryption schemes that equips two *multi-party protocols* KGen_t and Dec_t instead of algorithms KGen_p and Dec_p. Key generation protocol KGen_t is performed by n parties $P_1, \ldots, P_n$. At the end of this protocol, all parties agree on a public key PK, and each party P_i obtains its partial secret key sk_i. We denote the list of the partial secret keys as $\mathsf{SK} = (\mathsf{sk}_1, \ldots, \mathsf{sk}_n)$. Threshold decryption protocol Dec_t is performed by t or more parties. On input ciphertext CT as a public input and a partial secret key sk_i as a private input from P_i, it outputs the plaintext m.

IND-CCA and OW-CPA for (t, n)-threshold PKEs can be defined in the same way as IND-CCA and OW-CPA for regular PKE schemes. The difference is that the adversary can corrupt $t - 1$ parties of their own choice, which allows them to receive $t - 1$ partial secret keys and intermediate decryption results in response to decryption queries.

2.4 Key encapsulation mechanism

A key encapsulation mechanism (KEM) is defined as a triple of algorithms $\Pi_\mathrm{k} = (\mathsf{KGen}_\mathrm{k}, \mathsf{Encap}, \mathsf{Decap})$ with a session key space K_k and a ciphertext space C_k. $\mathsf{KGen}_\mathrm{k}(1^\lambda)$ generates a key pair $(\mathsf{pk}, \mathsf{sk})$. The encapsulation algorithm $\mathsf{Encap}(\mathsf{pk})$ outputs a session key $\mathsf{k} \in K_\mathrm{k}$ and its ciphertext $\mathsf{ct} \in C_\mathrm{k}$. The decapsulation algorithm $\mathsf{Decap}(\mathsf{sk}, \mathsf{ct})$ outputs a session key k. For correctness, $\mathsf{Decap}(\mathsf{sk}, \mathsf{ct}) = \mathsf{k}$ holds with overwhelming probability, if $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_\mathrm{k}(1^\lambda)$, $(\mathsf{k}, \mathsf{ct}) \leftarrow \mathsf{Encap}(\mathsf{pk})$.

For the security of KEM, we can define the indistinguishability of session keys, which guarantees that an adversary given a ciphertext cannot obtain any information about the session key. More precisely, an adversary $\mathcal{A}$ is first given a public key pk. Next $b \leftarrow_\$ \{0, 1\}$, $(\mathsf{ct}^*, \mathsf{k}_0) \leftarrow \mathsf{Encap}(\mathsf{pk})$ and $\mathsf{k}_1 \leftarrow_\$ K_\mathrm{k}$ are computed, and $(\mathsf{ct}^*, \mathsf{k}_b)$ is sent to $\mathcal{A}$. After that $\mathcal{A}$ is allowed to send $\mathsf{ct}(\neq \mathsf{ct}^*)$ to a decapsulation oracle O_{Decap} which returns $\mathsf{Decap}(\mathsf{sk}, \mathsf{ct})$. $\mathcal{A}$ outputs b' as the guessing bit of b. If the advantage of $\mathcal{A}$, defined by

$$\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\Pi_\mathrm{k},\mathcal{A}} := \left| 2 \cdot \Pr\left[b = b'\right] - 1 \right|,$$

is negligible for all QPT $\mathcal{A}$, we say Π_k is IND-CCA secure.

2.5 Symmetric-key encryption

A symmetric-key encryption (SKE) scheme is defined as a pair of algorithms $\Pi_\mathrm{s} = (\mathsf{Enc}_\mathrm{s}, \mathsf{Dec}_\mathrm{s})$ along with the key space K_s and the plaintext space M_s. For a randomly chosen symmetric key $\mathsf{k} \in K_\mathrm{s}$, a plaintext $\mathsf{m} \in M_\mathrm{s}$ is encrypted into a ciphertext ct as $\mathsf{ct} = \mathsf{Enc}_\mathrm{s}(\mathsf{k}, \mathsf{m})$. The ciphertext is decrypted by Dec_s as $\mathsf{m} = \mathsf{Dec}_\mathrm{s}(\mathsf{k}, \mathsf{ct})$. For correctness, $\mathsf{Dec}_\mathrm{s}(\mathsf{k}, \mathsf{Enc}_\mathrm{s}(\mathsf{k}, \mathsf{m})) = \mathsf{m}$ holds for all $\mathsf{k} \in K_\mathrm{s}$ and $\mathsf{m} \in M_\mathrm{s}$.

For the security notion of SKE, we give one-time IND-CCA and one-time IND-CPA [19] securities with the use of the hybrid construction in mind. The following game defines one-time IND-CCA security. First, a key is chosen randomly $\mathsf{k} \leftarrow_{\$} K_{\mathrm{s}}$. Next, the adversary $\mathcal{A}$ outputs a pair of plaintexts $\mathsf{m}_0^*, \mathsf{m}_1^*$, and receives $\mathsf{ct}^* := \mathsf{Enc}_{\mathrm{s}}(\mathsf{k}, \mathsf{m}_b^*)$, where b is a randomly-chosen challenge bit. After that $\mathcal{A}$ is allowed to send $\mathsf{ct}(\neq \mathsf{ct}^*)$ to a decryption oracle $O_{\mathsf{Dec}_{\mathrm{s}}}$ which returns $\mathsf{Dec}_{\mathrm{s}}(\mathsf{k}, \mathsf{ct})$ any number of times. Finally, $\mathcal{A}$ outputs $b' \in \{0, 1\}$. If $\mathcal{A}$'s advantage

$$\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\Pi_{\mathrm{s}},\mathcal{A}} := \left| 2 \cdot \Pr\left[b' = b\right] - 1 \right|$$

is negligibly small for any QPT adversary $\mathcal{A}$, then we say that Π_{s} is one-time IND-CCA secure. One-time IND-CPA security is exactly the same as one-time IND-CCA security, except that $\mathcal{A}$ is not allowed to access the decryption oracle. If the advantage $\mathcal{A}$ in this setting is negligibly small, then we say that Π_{s} is one-time IND-CPA secure.

2.6 Signature scheme

A signature scheme consists of three algorithms, $\Pi_{\mathrm{sig}} = (\mathsf{KGen}_{\mathrm{sig}}, \mathsf{Sign}, \mathsf{Verify})$. $\mathsf{KGen}_{\mathrm{sig}}$ takes the security parameter 1^λ as input and generates a public (verification) key vk and a secret (signing) key sigk. Sign takes a message m and a signing key sigk as input, and outputs a signature σ. Verify takes m, σ, and vk as input, and outputs 1 (accept) or 0 (reject). For all m and $(\mathsf{vk}, \mathsf{sigk}) \leftarrow \mathsf{KGen}_{\mathrm{sig}}(1^\lambda)$, $\Pr\left[\mathsf{Verify}(\mathsf{m}, \mathsf{Sign}(\mathsf{m}, \mathsf{sigk}), \mathsf{vk})) = 1\right] = 1$ must hold for correctness.

Signature schemes are often used in cryptographic systems as a building block. In such a case, one-time strong existential unforgeability against chosen message attack (one-time strong EUF-CMA) is required in general. Refer [4] for the definition of the one-time strong EUF-CMA security.

2.7 Dodis and Katz conversion

Dodis and Katz [24] proposed a generic construction of an IND-CCA secure threshold PKE from an IND-CCA secure (non-threshold) PKE using parallel encryption technique plus a secret sharing scheme and a one-time signature scheme. We roughly explain the (t, n)-threshold PKE converted from a (non-threshold) PKE with label[4] $\Pi_{\mathrm{p}} = (\mathsf{KGen}_{\mathrm{p}}, \mathsf{Enc}_{\mathrm{p}}, \mathsf{Dec}_{\mathrm{p}})$ as follows.

In the key generation process, each party P_i runs $\mathsf{KGen}_{\mathrm{p}}$ to generate a key pair $(\mathsf{pk}_i, \mathsf{sk}_i)$. In the encryption process, a plaintext m is first divided into n shares $\mathsf{s}_1, \ldots, \mathsf{s}_n$ using a (t, n)-secret sharing scheme, and a key pair $(\mathsf{vk}, \mathsf{sigk})$ of the one-time signature scheme is generated. Next, $\mathsf{ct}_i \leftarrow \mathsf{Enc}_{\mathrm{p}}(\mathsf{pk}_i, \mathsf{s}_i, \mathsf{vk})$ is computed for all $i \in [n]$, where vk is treated as a label. The ciphertext of m consists of the list $\mathsf{CT} := (\mathsf{ct}_1, \ldots, \mathsf{ct}_n)$, vk, and the signature $\sigma \leftarrow \mathsf{Sign}(\mathsf{sigk}, \mathsf{CT})$. In the decryption process, each P_i verifies the signature and computes $\mathsf{s}_i \leftarrow \mathsf{Dec}_{\mathrm{p}}(\mathsf{sk}_i, \mathsf{ct}_i, \mathsf{vk})$. Then P_i broadcasts the share s_i and reconstructs the plaintext m from the shares $\{\mathsf{s}_i\}_i$ of the secret sharing scheme.

The security of the Dodis–Katz conversion is stated in the following proposition.

[4] A PKE scheme with label can be easily constructed from any PKE schemes without label [44]. We denote a label as the last input of the encryption and decryption algorithms.

Proposition 1 [24, Theorem 1] *If the underlying PKE* Π_{p} *is* IND-CCA *secure and the signature scheme is a one-time strong* EUF-CMA *secure, then the* (t, n)*-threshold PKE from the Dodis–Katz conversion is* IND-CCA *secure.*

2.8 Cong et al. conversion

Cong et al. [16] proposed constructions of building IND-CCA secure PKEs from OW-CPA secure PKEs via the KEM-DEM paradigm. Their notable feature is that IND-CCA security of the resulting scheme is guaranteed even if the decryption result of KEM (i.e., the session key for DEM) is revealed. This feature has a great advantage in case the underlying OW-CPA PKE (used as a KEM) has an efficient threshold decryption protocol. Their hybrid construction has two versions: Hybrid_1 secure in the random oracle model (ROM), and Hybrid_2 secure in the quantum random oracle model (QROM) in which we are interested.

We recall the second conversion Hybrid_2. (Henceforth, CCMS conversion will refer to Hybrid_2 conversion.) Let $\Pi_{\mathrm{t}}^{\mathrm{ow}} = (\mathsf{KGen}_{\mathrm{t}}^{\mathrm{ow}}, \mathsf{Enc}_{\mathrm{t}}^{\mathrm{ow}}, \mathsf{Dec}_{\mathrm{t}}^{\mathrm{ow}})$ be a OW-CPA (t, n)-threshold PKE scheme with plaintext space M_{t}, $\Pi_{\mathrm{s}} = (\mathsf{Enc}_{\mathrm{s}}, \mathsf{Dec}_{\mathrm{s}})$ be an SKE scheme with plaintext space M_{s}, and key space K_{s}. Let

$$\mathsf{H} : M_{\mathrm{t}} \to K_{\mathrm{s}} \quad \mathsf{H}', \mathsf{H}'' : M_{\mathrm{t}} \to M_{\mathrm{t}}$$
$$\mathsf{G} : \{0,1\}^* \times M_{\mathrm{t}} \to \{0,1\}^{\ell_g}$$

be hash functions. The IND-CCA secure threshold PKE $\Pi_{\mathrm{t}} = (\mathsf{KGen}_{\mathrm{t}}, \mathsf{Enc}_{\mathrm{t}}, \mathsf{Dec}_{\mathrm{t}})$ from CCMS conversion is described as follows.

- $\mathsf{KGen}_{\mathrm{t}}(1^\lambda)$: It is identical to $\mathsf{KGen}_{\mathrm{t}}^{\mathrm{ow}}$. The output is $(\mathsf{SK}, \mathsf{PK}) \leftarrow \mathsf{KGen}_{\mathrm{t}}^{\mathrm{ow}}(1^\lambda)$.
- $\mathsf{Enc}_{\mathrm{t}}(\mathsf{PK}, \mathsf{m})$: On input a public key PK and plaintext $\mathsf{m} \in M_{\mathrm{s}}$, it computes the ciphertext $\mathsf{CT} = (\mathsf{ct}_1, \mathsf{ct}_2, \mathsf{ct}_3, \mathsf{ct}_4)$ as follows:

$$\begin{aligned} &k \leftarrow_{\$} M_{\mathrm{t}}, && \mathsf{k} \leftarrow \mathsf{H}(k), \quad \mu \leftarrow \mathsf{H}'(k) \\ &\mathsf{ct}_1 \leftarrow \mathsf{Enc}_{\mathrm{t}}^{\mathrm{ow}}(\mathsf{pk}, k), && \mathsf{ct}_2 \leftarrow \mathsf{Enc}_{\mathrm{s}}(\mathsf{k}, \mathsf{m}), \\ &\mathsf{ct}_3 \leftarrow \mathsf{G}(\mathsf{ct}_2, \mu), && \mathsf{ct}_4 \leftarrow \mathsf{H}''(k). \end{aligned}$$

- $\mathsf{Dec}_{\mathrm{t}}(\mathsf{SK}, \mathsf{CT})$: First, the parties jointly perform $\mathsf{Dec}_{\mathrm{t}}^{\mathrm{ow}}$ protocol to decrypt ct_1. After the protocol run, they have the secret share of k. Then, they evaluate $\mu \leftarrow \mathsf{H}'(k)$, $\mathsf{ct}_3 \leftarrow \mathsf{G}(\mathsf{ct}_2, \mu)$, $\mathsf{ct}_4 \leftarrow \mathsf{H}''(k)$, and check the validity of ct_3 and ct_4 by jointly performing the MPC for hash functions and the equality check. If the check does not pass, they output $\perp$ and abort the protocol. Otherwise, they obtain the reconstructed k. Finally, they compute $\mathsf{k} \leftarrow \mathsf{H}(k)$ and $\mathsf{m} \leftarrow \mathsf{Dec}_{\mathrm{s}}(\mathsf{k}, \mathsf{ct}_2)$ in the clear, and output (k, m).

The security of Π_{t} is stated as follows.

Proposition 2 [16, Theorem 3.2] $\mathsf{G}, \mathsf{H}, \mathsf{H}', \mathsf{H}''$ *are modeled as quantum random oracle. If* $\Pi_{\mathrm{t}}^{\mathrm{ow}}$ *is deterministic, rigid,* $\delta_\perp$*-*$\perp$*-aware,* δ_c*-collision free,* OW-CPA *secure, and has decryption failure probability* δ_f *for a randomly chosen plaintext and* Π_{s} *is (one-time)* IND-CPA *secure with negligibly small* $\delta_\perp, \delta_c, \delta_f$*, then* Π_{t} *is* IND-CCA *secure.*

Efficiency of the decryption protocol The decryption protocol mainly consists of three parts, the PKE decryption part, the validity check part, and the SKE decryption part. The efficiency of the first part depends on the underlying threshold PKE. The last part is lightweight because it is performed in the clear. As for the second part, Cong et al. suggested the use of an MPC-friendly hash function Rescue [3] as a hash function. Note that, although the input of G can

be long, the compression of ct_2 can be done in the normal way (without MPC), and only the compression of the very last part (distributed μ) needs to be done in a distributed fashion.

3 Code-based cryptography

In this section, we introduce the Classic McEliece as a typical example of code-based KEM. Further, we describe code-based signature schemes.

3.1 Goppa codes and Patterson decoding

First, we explain the binary Goppa code [30] on which Classic McEliece is based, and its decoding algorithm Patterson's algorithm [40] implemented in it.

Let $g(x) = \sum_{i=0}^{t_c} g_i x^i \in \mathbb{F}_{2^m}[x]$ be a monic irreducible polynomial of degree t_c called Goppa polynomial, and $\gamma = (\gamma_1, \ldots, \gamma_{n_c}) \in \mathbb{F}_{2^m}^{n_c}$ be n_c distinct supports such that $g(\gamma_i) \neq 0$. (k_c, n_c, t_c)-binary Goppa codes are defined by a parity-check matrix

$$H = \{h_{i,j}\} \in \mathbb{F}_{2^m}^{t_c \times n_c},$$
$$h_{i,j} = \frac{\gamma_j^{i-1}}{g(\gamma_j)} \ (i = 1, \ldots, t_c, \ j = 1, \ldots, n_c).$$

Since each element of the matrix H is in $\mathbb{F}_{2^m}$, H can be expressed as a $(t_c \cdot m) \times n_c$ matrix over $\mathbb{F}_2$. Denoting $k_c = n_c - m \cdot t_c$, it means H is a $(n_c - k_c) \times n_c$ binary matrix. In the following, we assume t_c is an even number.

Patterson's algorithm is a typical decoding algorithm for binary Goppa codes. For a given receiving word $v \in \{0, 1\}^{n_c}$, it recovers the error e if e's hamming weight $\mathrm{wt}(e)$ is at most t_c. The output e satisfies $Hv = He$ and $\mathrm{wt}(e) \leq t_c$. In Algorithm 1, we show a slightly modified version of Patterson's algorithm using the condition that t_c is an even number and $\mathrm{wt}(e) = t_c$.

3.2 Classic McEliece KEM

Classic McEliece KEM $\Pi_{\mathrm{k}}^{\mathrm{CM}}$ is an IND-CCA secure KEM based on the syndrome decoding problem [2]. Its core component is an OW-CPA secure public key encryption $\Pi_{\mathrm{p}}^{\mathrm{owCM}} = (\mathsf{KGen}_{\mathrm{p}}^{\mathrm{owCM}}, \mathsf{Enc}_{\mathrm{p}}^{\mathrm{owCM}}, \mathsf{Dec}_{\mathrm{p}}^{\mathrm{owCM}})$ described as follows. Its plaintext space is $W_{t_c,n_c} := \{e \in \mathbb{F}_2^{n_c} \mid \mathrm{wt}(e) = t_c\}$.

- $\mathsf{KGen}_{\mathrm{p}}^{\mathrm{owCM}}(1^\lambda)$: Generate a random parameter of (k_c, n_c, t_c)-binary Goppa code $g(x)$, γ and its parity check matrix $H = [I_{n_c-k_c} \mid H_{k_c}]$, where $I_{n_c-k_c}$ is the $(n_c - k_c) \times (n_c - k_c)$ identity matrix. Output $\mathsf{pk} := H_{k_c} \in \mathbb{F}_2^{(n_c-k_c)\times k_c}$ and $\mathsf{sk} := (g(x), \gamma)$.
- $\mathsf{Enc}_{\mathrm{p}}^{\mathrm{owCM}}(\mathsf{pk}, e)$: For a plaintext $e \in W_{t_c,n_c}$, output its ciphertext $c = [I_{n_c-k_c} \mid H_{k_c}] \cdot e \in \mathbb{F}_2^{n_c-k_c}$.
- $\mathsf{Dec}_{\mathrm{p}}^{\mathrm{owCM}}(\mathsf{sk}, c)$: For a given ciphertext $c \in \mathbb{F}_2^{n_c-k_c}$, set $v := [c \ 0 \ \ldots \ 0] \in \mathbb{F}_2^{n_c}$ by appending k_c zeros. Find e s.t. $Hv = He(= c)$ by using some decoding algorithm for the Goppa code. If $\mathrm{wt}(e) = t_c$ and $c = He$, output e, otherwise output $\perp$.

By using $\Pi_{\mathrm{p}}^{\mathrm{owCM}}$ and a hash function $\mathsf{H}_{\mathrm{cm}} : \{0, 1\} \times \{0, 1\}^{n_c} \times \{0, 1\}^{n_c-k_c} \to K_{\mathrm{k}}$, Classic McEliece KEM $\Pi_{\mathrm{k}}^{\mathrm{CM}} = (\mathsf{KGen}_{\mathrm{k}}^{\mathrm{CM}}, \mathsf{Encap}^{\mathrm{CM}}, \mathsf{Decap}^{\mathrm{CM}})$ is constructed as follows.

Algorithm 1 Patterson's algorithm.

Parameters: $g(x)$ is a Goppa polynomial,
$\gamma_1, \ldots, \gamma_{n_c}$ are supports,
$(x-\gamma_i)^{-1}$ is a polynomial that satisfies $(x-\gamma_i)^{-1}(x-\gamma_i) = 1 \bmod g(x)$,
T is the matrix representing the transformation $a(x) \to a(x)^2 \bmod g(x)$.

Input: $v = [v_1 \ \cdots \ v_{n_c}] \in \{0,1\}^{n_c}$
Output: $e = [e_1 \ \cdots \ e_{n_c}] \in \{0,1\}^{n_c}$

1: $s_v(x) := \sum_{i=1}^{n_c} v_i (x-\gamma_i)^{-1} \bmod g(x)$
2: $s'_v(x) := \frac{1}{s_v(x)} \bmod g(x)$
3: $u(x) := T^{-1}(x + s'_v(x))$ ▷ T^{-1} is the inversion matrix of T
4: Compute $\alpha(x)$ and $\beta(x)$ such that $s(x)g(x) + \beta(x)u(x) = \alpha(x)$ and $\deg(\alpha) = t_c/2$ ▷ Use the extended Euclidean algorithm
5: $\epsilon(x) := \alpha(x)^2 + x\beta(x)^2$
6: **for** $i \in [n_c]$ **do**
7: **if** $\epsilon(\gamma_i) = 0$ **then** $e_i := 1$
8: **else** $e_i := 0$
9: **end if**
10: **end for**
11: **return** $e := [e_1 \ \cdots \ e_{n_c}]$

- $\mathsf{KGen}_{\mathrm{k}}^{\mathrm{CM}}(1^\lambda)$: Generate $(\mathsf{sk}', \mathsf{pk}') \leftarrow \mathsf{KGen}_{\mathrm{p}}^{\mathrm{owCM}}(1^\lambda)$ and choose $s \leftarrow_{\$} \{0,1\}^{n_c}$. Output $\mathsf{sk} := (\mathsf{sk}', s)$, $\mathsf{pk} := \mathsf{pk}'$.
- $\mathsf{Encap}^{\mathrm{CM}}(\mathsf{pk})$: Randomly choose $e \leftarrow_{\$} W_{t_c,n_c}$. Compute $c \leftarrow \mathsf{Enc}_{\mathrm{p}}^{\mathrm{owCM}}(\mathsf{pk}, e)$ and $\mathsf{k} \leftarrow \mathsf{H}_{\mathrm{cm}}(1, e, c)$. Output a ciphertext $\mathsf{ct} := c$ and a session key k.
- $\mathsf{Decap}^{\mathrm{CM}}(\mathsf{sk}, \mathsf{ct})$: Set $c := \mathsf{ct}$. Compute $e \leftarrow \mathsf{Dec}_{\mathrm{p}}^{\mathrm{owCM}}(\mathsf{sk}, c)$. If $e \neq \perp$, output $\mathsf{k} \leftarrow \mathsf{H}_{\mathrm{cm}}(1, e, c)$. Otherwise output $\mathsf{k} \leftarrow \mathsf{H}_{\mathrm{cm}}(0, s, c)$.

$\Pi_{\mathrm{k}}^{\mathrm{CM}}$ is perfectly correct and rigid [2].

3.3 Code-based signature

In the literature, several code-based digital signature schemes have been proposed [8, 14, 17, 22, 26, 31, 45, 46]. Mainly, there are two approaches: hash-then-sign paradigm and a combination of proof-of-knowledge (PoK) and Fiat–Shamir transformation [27]. There are only a few hash-then-sign signature schemes. CFS signature [17] is the initial one. Its security is based on the hardness of the syndrome decoding problem, the same as the Classic McEliece. However, it has several problems such as choosing parameters, signature size, and signing algorithm complexity. Wave [22] is the state-of-the-art hash-then-sign-based code-based signature. It has a short signature (930 bytes for 128-bit security) but a large verification key (3.2 megabytes). On the other hand, a lot of PoK-based code-based signatures [8, 14, 26, 31, 45, 46] have been proposed. These schemes are derived from Stern's protocols [45], which are based on the syndrome decoding problem. PoK-based code-based signatures have short verification keys (about 100–200 bytes) but large signature sizes (about 15–30 kilobytes).

To our best knowledge, Sig 3 [8] has the shortest verification key plus signature size (165 plus 15,355 bytes) among signature schemes whose security relies on the syndrome decoding problem over $\mathbb{F}_2$. Moreover, it satisfies *strong* EUF-CMA security, which is required for the Dodis–Katz conversion.

4 Code-based threshold PKE from Dodis–Katz conversion

We can obtain an IND-CCA secure code-based threshold PKE by instantiating the Dodis–Katz conversion [24] with a code-based PKE and signature scheme. For completeness, we give a concrete description of the resulting scheme. We call it $\Pi_{\mathrm{t}}^{\mathrm{I}}$. We instantiate the building blocks in the Dodis–Katz conversion as follows:

- IND-CCA secure PKE with label: We use a PKE scheme derived from the Classic McEliece KEM $\Pi_{\mathrm{k}}^{\mathrm{CM}} = (\mathsf{KGen}_{\mathrm{k}}^{\mathrm{CM}}, \mathsf{Encap}^{\mathrm{CM}}, \mathsf{Decap}^{\mathrm{CM}})$ and IND-CCA secure SKE $\Pi_{\mathrm{s}} = (\mathsf{Enc}_{\mathrm{s}}, \mathsf{Dec}_{\mathrm{s}})$ via the KEM-DEM paradigm [44]. Let K_{s} be the key space of Π_{s}. To convert the PKE scheme into a scheme with label, a target collision resistance hash function $\mathsf{H}_\ell : \{0,1\}^* \to \{0,1\}^{\ell_{\mathrm{hash}}}$ is used.
- Secret sharing scheme: We use Krawczyk's scheme [35] with Shamir's scheme [41], in which the secret is encrypted with SKE, and the symmetric key of SKE is shared with Shamir's scheme.
- Strong EUF-CMA secure signature: We use the Sig 3 scheme $\Pi_{\mathrm{sig}}^{\mathrm{Sig3}} = (\mathsf{KGen}_{\mathrm{sig}}^{\mathrm{Sig3}}, \mathsf{Sign}^{\mathrm{Sig3}}, \mathsf{Verify}^{\mathrm{Sig3}})$ proposed by Bidoux et al. [8].

The first code-based threshold PKE scheme $\Pi_{\mathrm{t}}^{\mathrm{I}} = (\mathsf{KGen}_{\mathrm{t}}^{\mathrm{I}}, \mathsf{Enc}_{\mathrm{t}}^{\mathrm{I}}, \mathsf{Dec}_{\mathrm{t}}^{\mathrm{I}})$ is described as follows. Its plaintext space is M_{s}.

- $\mathsf{KGen}_{\mathrm{t}}^{\mathrm{I}}(1^\lambda)$: Each party P_i runs $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{KGen}_{\mathrm{k}}^{\mathrm{CM}}(1^\lambda)$ locally, and broadcasts pk_i. Set $\mathsf{PK} := (\mathsf{pk}_1, \ldots, \mathsf{pk}_n)$. sk_i is the partial secret key of P_i.
- $\mathsf{Enc}_{\mathrm{t}}^{\mathrm{I}}(\mathsf{PK}, \mathsf{m})$: Choose $\mathsf{k} \leftarrow_{\$} K_{\mathrm{s}}$. Let $(\mathsf{s}_1, \ldots, \mathsf{s}_n)$ be a set of Shamir's shares of k, and compute $\mathsf{ct}_0 \leftarrow \mathsf{Enc}_{\mathrm{s}}(\mathsf{k}, \mathsf{m})$. Generate a key pair $(\mathsf{vk}, \mathsf{sigk}) \leftarrow \mathsf{KGen}_{\mathrm{sig}}^{\mathrm{Sig3}}(1^\lambda)$. For all $i \in [n]$, compute $(\mathsf{ct}_{\mathrm{kem},i}, \mathsf{k}_i') \leftarrow \mathsf{Encap}^{\mathrm{CM}}(\mathsf{pk}_i)$, $\mathsf{ct}_{\mathrm{ske},i} \leftarrow \mathsf{Enc}_{\mathrm{s}}(\mathsf{k}_i', (\mathsf{H}_\ell(\mathsf{vk}), \mathsf{s}_i))$, and set $\mathsf{ct}_i := (\mathsf{ct}_{\mathrm{kem},i}, \mathsf{ct}_{\mathrm{ske},i})$. $\sigma \leftarrow \mathsf{Sign}^{\mathrm{Sig3}}(\mathsf{sigk}, (\mathsf{ct}_1, \ldots, \mathsf{ct}_n))$. Output $\mathsf{CT} := (\mathsf{ct}_0, \mathsf{ct}_1, \ldots, \mathsf{ct}_n, \mathsf{vk}, \sigma)$.
- $\mathsf{Dec}_{\mathrm{t}}^{\mathrm{I}}(\mathsf{SK}, \mathsf{CT})$: Check whether $\mathsf{Verify}^{\mathrm{Sig3}}(\mathsf{vk}, (\mathsf{ct}_1, \ldots, \mathsf{ct}_n), \sigma) = 1$. If not, output $\perp$. Otherwise, each party decrypts $\mathsf{k}_i' \leftarrow \mathsf{Decap}^{\mathrm{CM}}(\mathsf{sk}_i, \mathsf{ct}_{\mathrm{kem},i})$ and $(h_i, \mathsf{s}_i) \leftarrow \mathsf{Dec}_{\mathrm{s}}(\mathsf{k}_i', \mathsf{ct}_{\mathrm{ske},i})$. If $h_i \neq \mathsf{H}_\ell(\mathsf{vk})$, P_i outputs $\perp$. If no party outputs $\perp$, k is reconstructed from shares $\mathsf{s}_1, \ldots, \mathsf{s}_n$ and m is recovered as $\mathsf{m} \leftarrow \mathsf{Dec}_{\mathrm{s}}(\mathsf{k}, \mathsf{ct}_0)$.

From Proposition 1, the above scheme is IND-CCA secure.

5 Code-based threshold PKE from new conversion

In this section, we propose a new conversion for building an OW-CPA secure threshold PKE from a non-threshold OW-CPA secure PKE. Combining this new conversion and CCMS conversion, we obtain the second IND-CCA secure code-based threshold PKE, called $\Pi_{\mathrm{t}}^{\mathrm{II}}$, from a non-threshold OW-CPA secure PKE.

As for the new conversion, first, we show how to build an (n, n)-threshold PKE, named (n, n)-$\Pi_{\mathrm{t}}^{\mathrm{owII}}$, and then extend it into a (t, n)-threshold PKE. The scheme $\Pi_{\mathrm{t}}^{\mathrm{owII}}$ satisfies the requirements for applying CCMS conversion, so it can be converted into $\Pi_{\mathrm{t}}^{\mathrm{II}}$ with IND-CCA security.

5.1 Deterministic (n, n)-threshold PKE (n, n)-$\Pi_{\mathrm{t}}^{\mathrm{owI}}$

Our goal is to construct a new threshold PKE scheme that can be applied to CCMS conversion, that is, we want a scheme that is

- deterministic,
- rigid,
- $\delta_\perp$-$\perp$-aware for negligibly small $\delta_\perp$,
- δ_c-collision free for negligibly small δ_c,
- $(1-\delta_f)$-correct for negligibly small δ_f,
- OW-CPA secure.

At first glance, Dodis–Katz conversion gives a threshold PKE that has all these properties if we start with a non-threshold one that has all these properties. Unfortunately, this is not correct. Because a threshold secret sharing scheme used in Dodis–Katz conversion generates shares $(\mathsf{s}_1, \ldots, \mathsf{s}_n)$ probabilistically, the threshold PKE results in probabilistic. Note that, there is no deterministic threshold secret sharing scheme since a deterministically-computed share leaks some information about the secret.

We overcome this problem by changing the sharing method. Our construction utilizes a simple split to divide a plaintext into n shares[5] instead of a threshold secret-sharing scheme. Such a sharing method cannot be used to convert an IND-CCA scheme since each share leaks partial information of the shared secret. However, it is sufficient in our setting, as shown below. A plaintext m of the converted scheme is n times longer than that of the underlying PKE. That is, the plaintext space of $\Pi_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}$ is $M_{\mathrm{t}} = (M_{\mathrm{p}})^n$.

More concretely, our construction (n,n)-$\Pi_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}} = (\mathsf{KGen}_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}, \mathsf{Enc}_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}, \mathsf{Dec}_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}})$ is described as follows, where $\Pi_{\mathrm{p}}^{\mathrm{ow}} = (\mathsf{KGen}_{\mathrm{p}}^{\mathrm{ow}}, \mathsf{Enc}_{\mathrm{p}}^{\mathrm{ow}}, \mathsf{Dec}_{\mathrm{p}}^{\mathrm{ow}})$ is an underlying PKE.

- $\mathsf{KGen}_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}(1^\lambda)$: Each party P_i generates a key pair $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{KGen}_{\mathrm{p}}^{\mathrm{ow}}(1^\lambda)$ and broadcasts pk_i. Let the public key be $\mathsf{PK} := (\mathsf{pk}_1, \ldots, \mathsf{pk}_n)$, P_i's partial secret key be sk_i.
- $\mathsf{Enc}_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}(\mathsf{PK}, \bar{\mathsf{m}})$: The plaintext $\bar{\mathsf{m}} \in (M_{\mathrm{p}})^n$ is split as $\bar{\mathsf{m}} = (\mathsf{m}_1, \mathsf{m}_2, \ldots, \mathsf{m}_n)$. The ciphertext is $\mathsf{CT} = (\mathsf{ct}_1, \mathsf{ct}_2, \ldots, \mathsf{ct}_n)$, where $\mathsf{ct}_i = \mathsf{Enc}_{\mathrm{p}}^{\mathrm{ow}}(\mathsf{pk}_i, \mathsf{m}_i)$.
- $\mathsf{Dec}_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}(\mathsf{SK}, \mathsf{CT})$: Given $\mathsf{CT} = (\mathsf{ct}_1, \mathsf{ct}_2, \ldots, \mathsf{ct}_n)$, each P_i locally decrypts ct_i as $\mathsf{m}_i \leftarrow \mathsf{Dec}_{\mathrm{p}}^{\mathrm{ow}}(\mathsf{sk}_i, \mathsf{ct}_i)$, and broadcasts m_i. If $\mathsf{m}_i = \perp$ for some i, the protocol outputs $\perp$. Otherwise, the plaintext $\bar{\mathsf{m}} = (\mathsf{m}_1, \mathsf{m}_2, \ldots, \mathsf{m}_n)$ is output.

We first show (n,n)-$\Pi_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}$ is OW-CPA secure.

Theorem 1 *If $\Pi_{\mathrm{p}}^{\mathrm{ow}}$ is* OW-CPA *secure, then (n,n)-$\Pi_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}$ is* OW-CPA *secure.*

Proof We show that, if there exists an adversary $\mathcal{A}$ that breaks the OW-CPA security of (n,n)-$\Pi_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}$, there exists an adversary $\mathcal{B}$ that breaks the OW-CPA security of $\Pi_{\mathrm{p}}^{\mathrm{ow}}$ that uses $\mathcal{A}$ as a subroutine. Without loss of generality, we assume $\mathcal{A}$ corrupts the first $n-1$ parties. Let $I := \{1, \ldots, n-1\}$. $\mathcal{B}$ receives pk^* and $\mathsf{ct}^* (= \mathsf{Enc}_{\mathrm{p}}^{\mathrm{ow}}(\mathsf{pk}^*, \mathsf{m}^*))$ as an input. Note that m^* is chosen randomly from M_{p}. $\mathcal{B}$ generates $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{KGen}_{\mathrm{p}}^{\mathrm{ow}}(1^\lambda)$ for all $i \in I$, and sets $\mathsf{pk}_n := \mathsf{pk}^*$, and sends $\mathsf{PK} := (\mathsf{pk}_1, \ldots, \mathsf{pk}_{n-1}, \mathsf{pk}_n)$ and $(\mathsf{sk}_1, \ldots, \mathsf{sk}_{n-1})$ to $\mathcal{A}$. Next, $\mathcal{B}$ randomly chooses $\mathsf{m}_i \leftarrow_{\$} M_{\mathrm{p}}$ and computes $\mathsf{ct}_i = \mathsf{Enc}_{\mathrm{p}}^{\mathrm{ow}}(\mathsf{pk}_i, \mathsf{m}_i)$ for all $i \in I$. $\mathcal{B}$ sends $\mathsf{CT}^* := (\mathsf{ct}_1, \ldots, \mathsf{ct}_{n-1}, \mathsf{ct}^*)$ to $\mathcal{A}$ as the challenge ciphertext. The OW-CPA adversary $\mathcal{A}$ outputs $\bar{\mathsf{m}} = (\mathsf{m}_1, \mathsf{m}_2, \ldots, \mathsf{m}_n) \in (M_{\mathrm{p}})^n$. $\mathcal{B}$ outputs m_n as own output.

If $\mathcal{A}$ succeeds, $\mathsf{ct}^* = \mathsf{Enc}_{\mathrm{p}}^{\mathrm{ow}}(\mathsf{pk}^*, \mathsf{m}_n)$ holds. So, $\mathcal{B}$ also succeeds. However, since $\Pi_{\mathrm{p}}^{\mathrm{ow}}$ is OW-CPA secure, the advantage of $\mathcal{A}$ is negligible. Thus the advantage of $\mathcal{B}$ is also negligible, and thus (n,n)-$\Pi_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}$ is OW-CPA secure. □

We then show that (n,n)-$\Pi_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}$ satisfies the required properties for CCMS conversion.

[5] This method can be viewed as an extreme ramp scheme.

Theorem 2 *If $\Pi_{\mathrm{p}}^{\mathrm{ow}}$ is deterministic, rigid, δ_c-collision free, $\delta_\perp$-$\perp$-aware, $(1-\delta_f)$-correct, then (n,n)-$\Pi_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}$ is deterministic, rigid, δ'_c-collision free, $\delta'_\perp$-$\perp$-aware, and $(1-\delta'_f)$-correct, where $\delta'_c < n\delta_c$, $\delta'_\perp \leq n\delta_\perp$, $\delta'_f < n\delta_f$.*

Proof From the construction, (n,n)-$\Pi_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}$ is deterministic if $\Pi_{\mathrm{p}}^{\mathrm{ow}}$ is deterministic. Further, the other properties follow from Lemmas 1 to 4 below. □

Lemma 1 *If $\Pi_{\mathrm{p}}^{\mathrm{ow}}$ is rigid, then $\Pi_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}$ is also rigid.*

Proof Consider $\mathsf{CT} = (\mathsf{ct}_1, \ldots, \mathsf{ct}_n) \in C_{\mathrm{t}} \backslash C_{\mathrm{t}}^{\perp}$, where C_{t} is a ciphertext space of $\Pi_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}$. Then $\mathsf{Dec}_{\mathrm{p}}^{\mathrm{ow}}(\mathsf{sk}_i, \mathsf{ct}_i) \neq \perp$ (i.e., $\mathsf{ct}_i \in C_{\mathrm{p}} \backslash C_{\mathrm{p}}^{\perp}$) must hold for all i. (Otherwise, $\mathsf{Dec}_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}$ outputs $\perp$ by the definition of $\mathsf{Dec}_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}$.) Therefore, there exists $\mathsf{m}_i \in M_{\mathrm{p}}$ for all $i \in [n]$ such that $\mathsf{Dec}_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}(\mathsf{SK}, \mathsf{CT}) = (\mathsf{m}_1, \ldots, \mathsf{m}_n)$. Since $\Pi_{\mathrm{p}}^{\mathrm{ow}}$ is rigid, $\mathsf{Enc}_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}(\mathsf{PK}, \mathsf{Dec}_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}(\mathsf{SK}, \mathsf{CT})) = \mathsf{Enc}_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}(\mathsf{PK}, (\mathsf{m}_1, \ldots, \mathsf{m}_n)) = (\mathsf{Enc}_{\mathrm{p}}^{\mathrm{ow}}(\mathsf{pk}_1, \mathsf{m}_1), \ldots, \mathsf{Enc}_{\mathrm{p}}^{\mathrm{ow}}(\mathsf{pk}_n, \mathsf{m}_n)) = (\mathsf{ct}_1, \ldots, \mathsf{ct}_n) = \mathsf{CT}$ holds. □

Lemma 2 *If $\Pi_{\mathrm{p}}^{\mathrm{ow}}$ is δ_c-collision free, then $\Pi_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}$ is δ'_c-collision free for $\delta'_c < n\delta_c$.*

Proof It is clear that for fixed PK, two distinct plaintexts $\bar{\mathsf{m}} = (\mathsf{m}_1, \ldots, \mathsf{m}_n)$ and $\bar{\mathsf{m}}' = (\mathsf{m}'_1, \ldots, \mathsf{m}'_n)$ make a collision in $\Pi_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}$ if and only if there exists i such that $(\mathsf{m}_i, \mathsf{m}'_i)$ makes a collision in $\Pi_{\mathrm{p}}^{\mathrm{ow}}$ based on pk_i. Therefore, $\Pi_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}$ has a collision pair with probability $1-(1-\delta_c)^n < n\delta_c$. □

Lemma 3 *If $\Pi_{\mathrm{p}}^{\mathrm{ow}}$ is $\delta_\perp$-$\perp$-aware, then $\Pi_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}$ is $\delta'_\perp$-$\perp$-aware, where $\delta'_\perp \leq n\delta_\perp$.*

Proof Let $\mathcal{A}$ be an adversary of $\perp$-aware game for $\Pi_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}$. Consider following adversary $\mathcal{B}$ of $\perp$-aware game for $\Pi_{\mathrm{p}}^{\mathrm{ow}}$.

On input pk^*, $\mathcal{B}$ randomly chooses $i^* \in [n]$ and set $\mathsf{pk}_{i^*} := \mathsf{pk}^*$. For all $i (\neq i^*)$, $\mathcal{B}$ generates key pairs $(\mathsf{pk}_i, \mathsf{sk}_i)$, and runs $\mathcal{A}$ on input $\mathsf{PK} := (\mathsf{pk}_1, \ldots, \mathsf{pk}_n)$. If $\mathcal{A}$ outputs $\bar{\mathsf{m}} = (\mathsf{m}_1, \ldots, \mathsf{m}_n)$ and $\mathsf{CT} = (\mathsf{ct}_1, \ldots, \mathsf{ct}_n)$, $\mathcal{B}$ outputs m_{i^*} and ct_{i^*}.

We assume $\mathcal{A}$ wins $\perp$-aware game. In this case, the following holds:

$$\mathsf{CT} = \mathsf{Enc}_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}(\mathsf{PK}, \bar{\mathsf{m}}) \text{ and} \tag{1}$$

$$\mathsf{Dec}_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}(\mathsf{SK}, \mathsf{CT}) = \perp. \tag{2}$$

Equation (1) implies $\forall i : \mathsf{ct}_i = \mathsf{Enc}_{\mathrm{p}}^{\mathrm{ow}}(\mathsf{pk}_i, \mathsf{m}_i)$. Equation (2) implies there exists a non-empty set I such that $\mathsf{Dec}_{\mathrm{p}}^{\mathrm{ow}}(\mathsf{sk}_i, \mathsf{ct}_i) = \perp$ holds for $\forall i \in I$. Therefore, $\mathcal{B}$ wins the $\perp$-aware game for $\Pi_{\mathrm{p}}^{\mathrm{ow}}$ if $i^* \in I$, that occurs with probability at least $1/n$. □

Lemma 4 *If $\Pi_{\mathrm{p}}^{\mathrm{ow}}$ is $(1-\delta_f)$-correct, then $\Pi_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}$ is $(1-\delta'_f)$-correct for $\delta'_f < n\delta_f$.*

Proof $\mathsf{Dec}_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}(\mathsf{SK}, (\mathsf{ct}_1, \ldots, \mathsf{ct}_n))$ fails to decrypt, only if at least one of $\mathsf{Dec}_{\mathrm{p}}^{\mathrm{ow}}(\mathsf{sk}_i, \mathsf{ct}_i)$ $(i \in [n])$ fails to decrypt. Therefore, $\delta'_f = 1-(1-\delta_f)^n < n\delta_f$. □

This completes the proof of Theorem 2.

5.2 Deterministic (t, n)-threshold PKE $\Pi_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}$

We now extend (n,n)-$\Pi_{\mathrm{t}}^{\mathrm{ow}\mathbb{I}}$ into t-out-of-n setting for any $t \leq n$. The extension is realized by the replicated secret sharing scheme's technique to turn an (n,n)-threshold access structure

into a (t, n)-threshold one. More precisely, for each $(t-1)$-subgroup $G_j (\subset [n])$, a key pair $(\mathsf{sk}'_j, \mathsf{pk}'_j)$ is generated, and sk'_j is assigned to all parties $P_i \in \bar{G}_j$, where $\bar{G}_j := [n] \backslash G_j$. Totally, $N := \binom{n}{t-1}$ key pairs are generated, and each party is assigned $d = \binom{n-1}{t-2}$ secret keys sk'_j. If $t = 2$ or $t = n$, then $N = n$ holds. For example, in the $(2, 3)$-setting, there are three key pairs and each P_i is assigned two sk'_j.

The concrete description of (t, n)-$\Pi_{\mathrm{t}}^{\mathrm{ow\text{I\!I}}} = (\mathsf{KGen}_{\mathrm{t}}^{\mathrm{ow\text{I\!I}}}, \mathsf{Enc}_{\mathrm{t}}^{\mathrm{ow\text{I\!I}}}, \mathsf{Dec}_{\mathrm{t}}^{\mathrm{ow\text{I\!I}}})$ is as follows.

- $\mathsf{KGen}_{\mathrm{t}}^{\mathrm{ow\text{I\!I}}}(1^\lambda)$: For each $(t-1)$-subgroup G_j of $[n]$, parties in $\bar{G}_j$ jointly generate $(\mathsf{pk}'_j, \mathsf{sk}'_j) \leftarrow \mathsf{KGen}_{\mathrm{p}}^{\mathrm{ow}}(1^\lambda)$, and broadcast pk'_j. Set $\mathsf{PK} := (\mathsf{pk}'_1, \ldots, \mathsf{pk}'_N)$, where $N = \binom{n}{t-1}$. Each P_i sets $\mathsf{sk}_i := \{\mathsf{sk}'_j \mid P_i \in \bar{G}_j\}$.
- $\mathsf{Enc}_{\mathrm{t}}^{\mathrm{ow\text{I\!I}}}(\mathsf{PK}, \bar{\mathsf{m}})$: This is identical to $\mathsf{Enc}_{\mathrm{t}}^{\mathrm{ow\text{I\!I}}}$ of (N, N)-$\Pi_{\mathrm{t}}^{\mathrm{ow\text{I\!I}}}$. The ciphertext is $\mathsf{CT} = (\mathsf{ct}_1, \ldots, \mathsf{ct}_N)$.
- $\mathsf{Dec}_{\mathrm{t}}^{\mathrm{ow\text{I\!I}}}(\mathsf{SK}, \mathsf{CT})$: Each party P_i decrypts ct_j ($P_i \in \bar{G}_j$) by using $\mathsf{sk}'_j \in \mathsf{sk}_i$, and broadcasts the plaintext m_j. If t or more parties attend the protocol, m_j for all $j \in [N]$ are broadcast. If some of m_j is $\perp$, then the protocol outputs $\perp$, otherwise, $\bar{\mathsf{m}} := (\mathsf{m}_1, \ldots, \mathsf{m}_N)$ is obtained.

The security of (t, n)-$\Pi_{\mathrm{t}}^{\mathrm{ow\text{I\!I}}}$ is stated as follows.

Theorem 3 *If $\Pi_{\mathrm{p}}^{\mathrm{ow}}$ is* OW-CPA *secure, then (t, n)-$\Pi_{\mathrm{t}}^{\mathrm{ow\text{I\!I}}}$ is* OW-CPA *secure.*

Proof We show that if there exists an adversary $\mathcal{A}'$ that breaks OW-CPA security of (t, n)-$\Pi_{\mathrm{t}}^{\mathrm{ow\text{I\!I}}}$, then there exists an adversary $\mathcal{B}$ that breaks OW-CPA security of $\Pi_{\mathrm{p}}^{\mathrm{ow}}$.

Corrupting up to $t-1$ parties, $\mathcal{A}'$ can get at most $N-1$ secret keys sk'_j. So, $\mathcal{A}'$ is given the same knowledge of $\mathcal{A}$ in the proof of Theorem 1. Therefore, we can construct $\mathcal{B}$ that breaks OW-CPA of $\Pi_{\mathrm{p}}^{\mathrm{ow}}$ with the same argument as Theorem 1. □

We can show that (t, n)-$\Pi_{\mathrm{t}}^{\mathrm{ow\text{I\!I}}}$ satisfies the desired properties.

Theorem 4 *If $\Pi_{\mathrm{p}}^{\mathrm{ow}}$ is deterministic, rigid, δ_c-collision free, $\delta_\perp$-$\perp$-aware, and $(1-\delta_f)$-correct, then (t, n)-$\Pi_{\mathrm{t}}^{\mathrm{ow\text{I\!I}}}$ is deterministic, rigid, δ'_c-collision free, $\delta'_\perp$-$\perp$-aware, and $(1-\delta'_f)$-correct, where $\delta'_c < N\delta_c$, $\delta'_\perp \leq N\delta_\perp$, $\delta'_f < N\delta_f$.*

Proof The proof is identical to the proof of Theorem 2. □

5.3 Second IND-CCA secure scheme $\Pi_{\mathrm{t}}^{\mathrm{I}}$

Let $\Pi_{\mathrm{p}}^{\mathrm{owCM}}$ be the underlying PKE of $\Pi_{\mathrm{t}}^{\mathrm{ow\text{I\!I}}}$. Then $\Pi_{\mathrm{t}}^{\mathrm{ow\text{I\!I}}}$ satisfies all requirements for CCMS conversion, and it can be converted into an IND-CCA one. We slightly modify CCMS conversion, because $\Pi_{\mathrm{t}}^{\mathrm{ow\text{I\!I}}}$'s plaintext space $(M_{\mathrm{p}})^N = (W_{t_c,n_c})^N \subset \mathbb{F}_2^{Nn_c}$ is a sparse set. More precisely, we change the ranges of H' and H'' and the domain of the second input of G from $(W_{t_c,n_c})^N$ to $\{0,1\}^{N\ell_h}$ for some ℓ_h that satisfies $2^{\ell_h} \geq |W_{t_c,n_c}|$. With such modification, Proposition 2 can be proven in the same way, except $|M_{\mathrm{p}}|$ is replaced with $2^{N\ell_h}$.[6] So, our construction uses the following hash functions:

$$\mathsf{H} : (W_{t_c,n_c})^N \to K_{\mathrm{s}},$$
$$\mathsf{H}', \mathsf{H}'' : (W_{t_c,n_c})^N \to \{0,1\}^{N\ell_h},$$

[6] In the security proof, $\mathsf{H}''(\cdot)$ is treated as $\tilde{\mathsf{H}}(\mathsf{encode}(\cdot))$ for some injective mapping $\mathsf{encode} : (W_{t_c,n_c})^N \to \{0,1\}^{N\ell_h}$ and a hash function $\tilde{\mathsf{H}} : \{0,1\}^{N\ell_h} \to \{0,1\}^{N\ell_h}$.

$$\mathsf{G} : \{0,1\}^* \times \{0,1\}^{N\ell_h} \to \{0,1\}^{\ell_g}.$$

$\Pi_{\mathrm{t}}^{\mathrm{II}} = (\mathsf{KGen}_{\mathrm{t}}^{\mathrm{II}}, \mathsf{Enc}_{\mathrm{t}}^{\mathrm{II}}, \mathsf{Dec}_{\mathrm{t}}^{\mathrm{II}})$ is described as follows.

- $\mathsf{KGen}_{\mathrm{t}}^{\mathrm{II}}(1^\lambda)$: Parties execute $\mathsf{KGen}_{\mathrm{t}}^{\mathrm{owII}}(1^\lambda)$. The public key is $\mathsf{PK} = (\mathsf{pk}'_1, \ldots, \mathsf{pk}'_N)$, and P_i has several sk'_i as a partial secret key.
- $\mathsf{Enc}_{\mathrm{t}}^{\mathrm{II}}(\mathsf{PK}, \mathsf{m})$: First $\bar{k} := (k_1, \ldots, k_N) \leftarrow_{\$} (W_{t_c,n_c})^N$ is chosen randomly. Next $\mathsf{k} \leftarrow \mathsf{H}(\bar{k})$, $\mu \leftarrow \mathsf{H}'(\bar{k})$, $\mathsf{ct}_{1,j} \leftarrow \mathsf{Enc}_{\mathrm{p}}^{\mathrm{owCM}}(\mathsf{pk}'_j, k_j)$ $(j \in [N])$, $\mathsf{ct}_2 \leftarrow \mathsf{Enc}_{\mathrm{s}}(\mathsf{k}, \mathsf{m})$, $\mathsf{ct}_3 \leftarrow \mathsf{G}(\mathsf{ct}_2, \mu)$, $\mathsf{ct}_4 \leftarrow \mathsf{H}''(\bar{k})$ are computed. Output $\mathsf{CT} = (\mathsf{ct}_{1,1}, \ldots, \mathsf{ct}_{1,N}, \mathsf{ct}_2, \mathsf{ct}_3, \mathsf{ct}_4)$.
- $\mathsf{Dec}_{\mathrm{t}}^{\mathrm{II}}(\mathsf{SK}, \mathsf{CT})$: For $\mathsf{CT} = (\mathsf{ct}_{1,1}, \ldots, \mathsf{ct}_{1,N}, \mathsf{ct}_2, \mathsf{ct}_3, \mathsf{ct}_4)$, each P_i decrypts $\mathsf{ct}_{1,j}$ $(P_i \in \bar{G}_j)$ as $k_j \leftarrow \mathsf{Dec}_{\mathrm{p}}^{\mathrm{owCM}}(\mathsf{sk}'_j, \mathsf{ct}_{1,j})$. For each $j \in [N]$, one of $P_i \in \bar{G}_j$ (who knows k_j) distributes k_j by using a (t, n)-threshold secret sharing scheme. At this moment, $\bar{k} = (k_1, \ldots, k_N)$ is shared among all parties. Next, $\mu \leftarrow \mathsf{H}'(\bar{k})$, $\widetilde{\mathsf{ct}}_3 \leftarrow \mathsf{G}(\mathsf{ct}_2, \mu)$ and $\widetilde{\mathsf{ct}}_4 \leftarrow \mathsf{H}''(\bar{k})$ are computed secretly by using MPC for hash functions. Then equality $\widetilde{\mathsf{ct}}_3 = \mathsf{ct}_3$ and $\widetilde{\mathsf{ct}}_4 = \mathsf{ct}_4$ are checked. If the check does not pass, $\perp$ is output and the protocol is aborted. Otherwise, $\bar{k}$ is reconstructed. $\mathsf{k} \leftarrow \mathsf{H}(\bar{k})$ and $\mathsf{m} \leftarrow \mathsf{Dec}_{\mathrm{s}}(\mathsf{k}, \mathsf{ct}_2)$ are locally computed.

6 Code-based threshold PKE from new multi-party decryption protocol

The third code-based threshold PKE is constructed using the same approach Cong et al. used to construct lattice-based threshold PKEs [16]. We design a practical MPC for computing the decryption algorithm of the OW-CPA secure Classic McEliece PKE $\Pi_{\mathrm{p}}^{\mathrm{owCM}}$, and then convert it to IND-CCA one applying CCMS conversion.

Recall that the decryption algorithm consists of three steps: (i) append k_c zeros to the ciphertext c, (ii) find e such that $Hv = He = c$ by following Patterson's decoding algorithm shown in Algorithm 1, and (iii) check if $\mathrm{wt}(e) = t_c$ and $c = He$. Steps (i) and (iii) are easily computed by MPCs. On the other hand, Patterson's decoding algorithm in step (ii) includes processes for which efficient MPCs cannot be trivially constructed (Lines 2 and 4). So, we first describe how to design an MPC for Patterson's decoding algorithm, especially focusing on Lines 2 and 4 (in Sect. 6.1). Next, we explain the MPC for the whole decryption algorithm of $\Pi_{\mathrm{p}}^{\mathrm{owCM}}$ including steps (i) and (iii) (in Sect. 6.2). Finally, we give the OW-CPA secure code-based threshold PKE $\Pi_{\mathrm{t}}^{\mathrm{owIII}}$ based on this MPC and the IND-CCA one $\Pi_{\mathrm{t}}^{\mathrm{III}}$ obtained by applying CCMS conversion in Sect. 6.3.

6.1 MPC for Patterson decoding

Of Algorithm 1 (Patterson Decoding), all procedures except Lines 2 and 4 are easily computed by MPCs by using the existing MPC protocols for the equivalence decision with zero [11] and the assignment calculation [20]. So, we first consider concrete (MPC-friendly) procedures for the non-trivial parts Lines 2 and 4.

In both Lines 2 and 4 (Algorithm 1), it solves the following same problem; for given two polynomials $g(x)$, $f(x)$ and the target degree d, find (some of) polynomials $a(x), b(x), c(x)$ such that $a(x)g(x) + b(x)f(x) = c(x)$ and $\deg(c) = d$. To solve this problem, instead of the Euclidean algorithm which is not MPC-friendly, we use a new procedure based on the idea used in [36] to design the MPC for a greatest common divisor (GCD) of polynomials. Ours is much simpler than theirs because the degree of c is fixed, i.e., $\deg(c) = 0$ in Line

2 and $\deg(c) = t_c/2$ in Line 4. Now, we introduce the definition and some properties of a subresultant matrix used in [36], then we describe our procedure.

Let $g(x) = \sum_{j=0}^{t_c} g_j x^j$ and $f(x) = \sum_{j=0}^{u} f_j x^j$ be polynomials over $\mathbb{F}_{2^m}$ of degree t_c and u ($u < t_c$), respectively. The subresultant matrix S_i ($i \in [u]$) is defined by a $(t_c + u - 2i) \times (t_c + u - 2i)$ matrix of the form:

$$S_i := \begin{bmatrix} g_{t_c} & & & f_u & & \\ \vdots & \ddots & & \vdots & \ddots & \\ g_{t_c-u+i+1} & \cdots & g_{t_c} & \vdots & & \ddots \\ \vdots & & \vdots & f_{u-t_c+i+1} & \cdots\cdots & f_u \\ \vdots & & \vdots & \vdots & & \vdots \\ g_{2i-u+1} & \cdots & g_i & f_{2i-t_c+1} & \cdots\cdots & f_i \end{bmatrix},$$

where $g_j = f_j = 0$ ($j < 0$). For this matrix, the following two propositions hold.

Proposition 3 [47] *If a polynomial $c(x)$ of degree i appears as a remainder polynomial during the computation of the extended Euclidean algorithm on input $g(x)$ and $f(x)$, $\det(S_i) \neq 0$ holds. Otherwise, $\det(S_i) = 0$.*

Proposition 4 [47] *If $\det(S_i) \neq 0$, the linear system $S_i \cdot y^T = [0 \ \ldots \ 0 \ 1]^T$* [7] *has a unique solution $y = [y_1 \ \ldots \ y_{u+t_c-2i}]$, and $a(x) := y_1 x^{u-i-1} + \cdots + y_{u-i}$ and $b(x) := y_{u-i+1} x^{t_c-i-1} + \cdots + y_{u+t_c-2i}$ satisfy $a(x)g(x) + b(x)f(x) = c(x)$.*

These propositions also work for polynomial $f(x)$ whose exact degree u is unknown but at most $t_c - 1$.

Theorem 5 *Let S_i be the subresultant matrix constructed from $g(x) = \sum_{j=0}^{t_c} g_j x^j$ and $f(x) = \sum_{j=0}^{u} f_j x^j$ ($u \leq t_c - 1$) and let $\bar{S}_i$ be the one from $g(x)$ and $f(x) = \sum_{j=0}^{t_c-1} f_j x^j$ ($f_{t_c-1}, \ldots, f_{u+1} = 0$). Then, the following statements hold.*

1. $\det(\bar{S}_i) \neq 0$ *if and only if* $\det(S_i) \neq 0$.
2. *The polynomials obtained from $\bar{y}^T = \bar{S}_i^{-1} \times [0 \ \ldots \ 0 \ 1]^T$ and the one form $y^T = S_i^{-1} \times [0 \ \ldots \ 0 \ 1]^T$ is equivalent.*

Proof From the definition of a subresultant matrix, we have

$$\bar{S}_i = \begin{bmatrix} A & O \\ B & S_i \end{bmatrix},$$

$$A = \begin{bmatrix} g_{t_c} & & \\ \vdots & \ddots & \\ g_{u-2} & \cdots & g_{t_c} \end{bmatrix}, B = \begin{bmatrix} g_{u-3} & \cdots & g_{t_c-1} \\ \vdots & & \vdots \\ g_{2i-t_c+2} & \cdots & g_{2i-u} \end{bmatrix},$$

where O is a zero matrix. As the matrix A is a lower triangular matrix, A is invertible. Therefore, from the property of a partitioned matrix, we have that if $\det(S_i) \neq 0$, then

$$\bar{S}_i^{-1} = \begin{bmatrix} A^{-1} & O \\ -S_i^{-1} B A^{-1} & S_i^{-1} \end{bmatrix},$$

[7] a^T is a transposed vector of a.

and if $\det(S_i) = 0$, then $\det(\bar{S}_i) = 0$. Moreover, the value $\bar{y}^T = \bar{S}_i^{-1} \times [0 \ \ldots \ 0 \ 1]^T$ is value of $t_c - u - 1$ zeros connected to the left of $y^T = S_i^{-1} \times [0 \ \ldots \ 0 \ 1]^T$. Thus, the two polynomials $a(x)$ and $b(x)$ obtained from $\bar{y}$ are equivalent to the ones from y. □

From Theorem 5, we can define the subresultant matrix assuming $\deg(f) = t_c - 1$ even if the exact degree may be $\deg(f) < t_c - 1$. Thus, we can use Propositions 3 and 4 and thus we can describe the concrete procedure of Line 2 as follows.

2-1: Construct S_0 from $g(x)$ and $f(x) := s_v(x)$.
2-2: Compute the inverse matrix S_0^{-1}.
2-3: Compute $y^T = S_0^{-1} \cdot [0 \ \ldots \ 0 \ 1]^T = [y_1 \ \ldots \ y_{2t_c-1}]^T$.
2-4: Output $s'_v(x) := b(x) = y_{t_c-1}x^{t_c} + \cdots + y_{2t_c-1}$.

Similarly, the procedure of Line 4 is described as follows.

4-1: Construct $S_{t_c/2}$ from $g(x)$ and $f(x) := u(x)$.
4-2: Compute the inverse matrix $S_{t_c/2}^{-1}$.
4-3: Compute $y^T = S_{t_c/2}^{-1} \cdot [0 \ \ldots \ 0 \ 1]^T = [y_1 \ \ldots \ y_{t_c-1}]^T$.
4-4: Let $a(x) = y_1 x^{t_c/2-2} + \cdots + y_{t_c/2-1}$ and $b(x) = y_{t_c/2}x^{t_c/2-1} + \cdots + y_{t_c-1}$.
4-5: Output $\beta(x) := b(x)$ and $\alpha(x) := c(x) = a(x) \cdot g(x) + b(x) \cdot f(x)$.

We now show the MPC for Patterson's algorithm. Assume that parties share a Goppa polynomial $g(x)$, supports $\gamma_1, \ldots, \gamma_{n_c}$, polynomials $(x - \gamma_i)^{-1}$ determined by supports, and the matrix T^{-1} determined by $g(x)$.[8] A word $v \in \{0, 1\}^{n_c}$ is given to each party as input. We also assume that a share of a polynomial is expressed as shares of its coefficients over $\mathbb{F}_{2^m}$. All these shares are generated in $\mathbb{F}_{2^m}$. By using the procedures above, Lines 2 and 4 in Algorithm 1 can be evaluated in a distributed fashion with MPCs for inverting matrix [5] and polynomial multiplication [36]. Since other lines in Algorithm 1 also can be computed distributedly, we obtain an MPC for Patterson's decoding algorithm.

The cost of the MPC for Patterson's algorithm is $O(t_c^3 + n_c t_c)$ times the invocation of an MPC for multiplication. It is derived from the cost of the MPCs for inverting matrix in Lines 2 and 4 and the assignment calculation in Line 6 on Algorithm 1. We note that at the end of the MPC, the output (i.e., error vector) is shared among the parties.

6.2 MPC for decryption of OW-CPA classic McEliece PKE

We now provide an MPC for the decryption of OW-CPA secure Classic McEliece PKE. Given a ciphertext c, the parties can jointly perform $\mathsf{Dec}_{\mathrm{p}}^{\mathrm{owCM}}(\mathsf{sk}, c)$ as follows.

1. Each party P_i sets $v := [c \ 0 \ \ldots \ 0]$ by appending k_c zeros.
2. Parties perform the MPC for Patterson's algorithm shown in Sect. 6.1 on input v and secret key's share sk_i, and obtain shared error vector $e = (e_1, \ldots, e_{n_c})$. (Each e_i is 0 or 1, but is shared as an element of $\mathbb{F}_{2^m}$.)
3. Parties jointly compute $w := \mathrm{wt}(e) = \sum_{i=1}^{n_c} e_i$. In this summation, each addition is done by an MPC that simulates a full-adder circuit, but each XOR gate and AND gate are evaluated by the MPC for addition (i.e., local computation) and multiplication in $\mathbb{F}_{2^m}$, respectively.
   As the result, each party obtains shares of $w_1, \ldots, w_{n'_c}$ where $(w_{n'_c}, \ldots, w_1)_2$ is the binary representation of w and $n'_c = \lceil \log_2 n_c \rceil + 1$.

[8] By sharing $(x - \gamma_i)^{-1}$ and T^{-1} in advance, we can save n_c invocations of the MPC for inverting matrix and compute a square root by an MPC for matrix multiplication.

4. Parties jointly compute $o_1 := \prod_{i=1}^{n'_c}(w_i - t_i + 1)$, where $(t_{n'_c}, \ldots, t_1)_2$ is the binary representation of t_c. Clearly, $o_1 = 1$ if $\mathrm{wt}(e) = t_c$, $o_1 = 0$ otherwise.
5. Each party compute the share of $e' := He - c = (e'_1, \ldots, e'_{n_c})$ in $\mathbb{F}_{2^m}$. Note that only e is shared, so this can be done by linear combinations. In addition, all components of H, e, c are 0 or 1, so e'_i are also 0 or 1.
6. Parties jointly compute $o_2 := \prod_{i=1}^{n_c}(e'_i + 1)$. Clearly, $o_2 = 1$ if $He = c$, $o_2 = 0$ otherwise.
7. Finally parties compute $o_1 \times o_2$ by running the MPC and reconstruct the product. If the result is 0, $\perp$ is output, otherwise, reconstruct e and output it.

The number of invocations of an MPC for multiplication in the MPCs other than Patterson's algorithm is $O(n_c)$. Therefore, the total cost of the MPC for the decryption mainly comes from evaluating Patterson's algorithm i.e. $O(t_c^3 + n_c t_c)$ as above.

6.3 Third IND-CCA secure scheme $\Pi_{\mathrm{t}}^{\blacksquare}$

Using the MPC for decryption of $\Pi_{\mathrm{p}}^{\mathrm{owCM}}$ shown in Sect. 6.2, we obtain the threshold version of $\Pi_{\mathrm{p}}^{\mathrm{owCM}}$. We call it $\Pi_{\mathrm{t}}^{\mathrm{owIII}} = (\mathsf{KGen}_{\mathrm{t}}^{\mathrm{owIII}}, \mathsf{Enc}_{\mathrm{t}}^{\mathrm{owIII}}, \mathsf{Dec}_{\mathrm{t}}^{\mathrm{owIII}})$.

- $\mathsf{KGen}_{\mathrm{t}}^{\mathrm{owIII}}(1^\lambda)$: The parties run the MPC to generate random parameters of (k_c, n_c, t_c)-binary Goppa codes $g(x)$, $\gamma = \{\gamma_i\}_i$ and it's parity check matrix $H = [I_{n_c - k_c} \mid H_{k_c}]$, the inverse matrix T^{-1} and polynomials $(x - \gamma_i)^{-1} \bmod g(x)$. The public key is $\mathsf{PK} = H_{k_c}$ and P_i's partial secret key sk_i consists of shares of $g(x)$, $\{\gamma_i\}_i$, T^{-1}, $\{(x - \gamma_i)^{-1}\}_i$.
- $\mathsf{Enc}_{\mathrm{t}}^{\mathrm{owIII}}(\mathsf{PK}, e)$:This is identical to $\mathsf{Enc}_{\mathrm{p}}^{\mathrm{owCM}}$. The ciphertext of $e \in W_{t_c, n_c}$ is $c := He$.
- $\mathsf{Dec}_{\mathrm{t}}^{\mathrm{owIII}}(\mathsf{SK}, c)$: Parties run the MPC shown in Sect. 6.2 and get e that satisfies $He = c$.

We do not describe the details of the key generation protocol, because any algorithm can be evaluated in a distributed fashion using MPCs theoretically. The key generation protocol is performed only once, so some degree of inefficiency is not a practical problem.

We can show $\Pi_{\mathrm{t}}^{\mathrm{owIII}}$ is OW-CPA secure and takes over the properties of $\Pi_{\mathrm{p}}^{\mathrm{owCM}}$.

Theorem 6 *If $\Pi_{\mathrm{p}}^{\mathrm{owCM}}$ is OW-CPA secure, then the threshold PKE $\Pi_{\mathrm{t}}^{\mathrm{owIII}}$ is OW-CPA secure.*

Proof The partial secret key of each party is all share values. Further, its key-generation and decryption protocols $\mathsf{KGen}_{\mathrm{t}}^{\mathrm{owIII}}$ and $\mathsf{Dec}_{\mathrm{t}}^{\mathrm{owIII}}$ leak no additional information to parties other than they should know (i.e., P_i learns only PK and its sk_i during $\mathsf{KGen}_{\mathrm{t}}^{\mathrm{owIII}}$, while the decrypted plaintext m during $\mathsf{Dec}_{\mathrm{t}}^{\mathrm{owIII}}$). Therefore, it is easy to construct an adversary algorithm $\mathcal{B}$ of OW-CPA for $\Pi_{\mathrm{p}}^{\mathrm{owCM}}$ by using an adversary algorithm $\mathcal{A}$ of OW-CPA for $\Pi_{\mathrm{t}}^{\mathrm{owIII}}$ as a subroutine. Therefore, OW-CPA security of $\Pi_{\mathrm{p}}^{\mathrm{owCM}}$ implies OW-CPA security of $\Pi_{\mathrm{t}}^{\mathrm{owIII}}$. □

Theorem 7 *If the MPCs in the threshold decryption do not fail, $\Pi_{\mathrm{t}}^{\mathrm{owIII}}$ is perfectly correct and rigid.*

Proof Given a valid ciphertext $c = He$ for some plantext $e \in W_{t_c, n_c}$, Patterson's algorithm recovers the error vector e with probability 1. Also, no failure happens during the MPCs from the assumption. Therefore, $\Pi_{\mathrm{t}}^{\mathrm{owIII}}$ is perfectly correct. Thus $\mathsf{Dec}_{\mathrm{t}}^{\mathrm{owIII}}(\mathsf{SK}, c) = e$ always holds, and we have $\mathsf{Enc}_{\mathrm{t}}^{\mathrm{owIII}}(\mathsf{PK}, \mathsf{Dec}_{\mathrm{t}}^{\mathrm{owIII}}(\mathsf{SK}, c)) = c$ because $\Pi_{\mathrm{t}}^{\mathrm{owIII}}$ is deterministic. Therefore, $\Pi_{\mathrm{t}}^{\mathrm{owIII}}$ is rigid. □

Theorem 7 implies that CCMS conversion is adaptable to $\Pi_{\mathrm{t}}^{\mathrm{owIII}}$. Thus, we obtain the IND-CCA secure threshold PKE, called $\Pi_{\mathrm{t}}^{\mathrm{III}} = (\mathsf{KGen}_{\mathrm{t}}^{\mathrm{III}}, \mathsf{Enc}_{\mathrm{t}}^{\mathrm{III}}, \mathsf{Dec}_{\mathrm{t}}^{\mathrm{III}})$, described as follows. Its plaintext space is M_{s}.

- $\mathsf{KGen}_{\mathrm{t}}^{\mathrm{III}}(1^\lambda)$: The parties run $\mathsf{KGen}_{\mathrm{t}}^{\mathrm{owIII}}(1^\lambda)$. The public key $\mathsf{PK} = H_{k_c}$ determines $H = [I_{n_c-k_c} \mid H_{k_c}]$.
- $\mathsf{Enc}_{\mathrm{t}}^{\mathrm{III}}(\mathsf{PK}, \mathsf{m})$: First $e \in W_{t_c,n_c}$ is chosen randomly. Next $\mathsf{k} \leftarrow \mathsf{H}(e)$, $\mu \leftarrow \mathsf{H}'(e)$, $\mathsf{ct}_1 \leftarrow He$, $\mathsf{ct}_2 \leftarrow \mathsf{Enc}_{\mathrm{s}}(\mathsf{k}, \mathsf{m})$, $\mathsf{ct}_3 \leftarrow \mathsf{G}(\mathsf{ct}_2, \mu)$, $\mathsf{ct}_4 \leftarrow \mathsf{H}''(e)$ are computed. Output $\mathsf{CT} = (\mathsf{ct}_1, \mathsf{ct}_2, \mathsf{ct}_3, \mathsf{ct}_4)$.
- $\mathsf{Dec}_{\mathrm{t}}^{\mathrm{III}}(\mathsf{SK}, \mathsf{CT})$: For $\mathsf{CT} = (\mathsf{ct}_1, \mathsf{ct}_2, \mathsf{ct}_3, \mathsf{ct}_4)$, parties first execute $\mathsf{Dec}_{\mathrm{t}}^{\mathrm{owIII}}(\mathsf{SK}, \mathsf{ct}_1)$. Note that the result e is shared at this moment. If $\perp$ is output, they abort the protocol and output $\perp$. Otherwise, $\mu \leftarrow \mathsf{H}'(e)$, $\widetilde{\mathsf{ct}}_3 \leftarrow \mathsf{G}(\mathsf{ct}_2, \mu)$ and $\widetilde{\mathsf{ct}}_4 \leftarrow \mathsf{H}''(e)$ are computed, equality $\widetilde{\mathsf{ct}}_3 = \mathsf{ct}_3$ and $\widetilde{\mathsf{ct}}_4 = \mathsf{ct}_4$ are checked by using MPCs. If the check does not pass, $\perp$ is output and the protocol is aborted. Otherwise, e is reconstructed. $\mathsf{k} \leftarrow \mathsf{H}(e)$ and $\mathsf{m} \leftarrow \mathsf{Dec}_{\mathrm{s}}(\mathsf{k}, \mathsf{ct}_2)$ are locally computed.

From Proposition 2, $\Pi_{\mathrm{t}}^{\mathrm{III}}$ is IND-CCA secure in the QROM.

7 Efficiency comparison

In this section, we compare three IND-CCA secure code-based threshold PKEs $\Pi_{\mathrm{t}}^{\mathrm{I}}$, $\Pi_{\mathrm{t}}^{\mathrm{II}}$ and $\Pi_{\mathrm{t}}^{\mathrm{III}}$ according to their ciphertext size and communication complexity of threshold decryption. We instantiate the components used in $\Pi_{\mathrm{t}}^{\mathrm{I}}$, $\Pi_{\mathrm{t}}^{\mathrm{II}}$ and $\Pi_{\mathrm{t}}^{\mathrm{III}}$ with the parameters for 128-bit security as follows.

- Classic McEliece $\Pi_{\mathrm{k}}^{\mathrm{CM}}$ and $\Pi_{\mathrm{p}}^{\mathrm{owCM}}$: we use the parameter set `kem/mceliece348864` in [2].
- Signature scheme $\Pi_{\mathrm{sig}}^{\mathrm{Sig3}}$: we use the 128-bit security parameter based on the hardness of the syndrome decoding problem in [8].
- Hash functions H_ℓ, H_{cm} and H: we use SHAKE-256.
- Hash functions H' and H'': we use an MPC-friendly hash Vision[9] [3]. As the parameters of Vision, the bit-size of the base field n_v, the number of field elements in the state m_v, capacity c_v, and rate r_v, we set $(n_{v_1}, m_{v_1}, c_{v_1}, r_{v_1}) = (12, 12, 2, 10)$.
- Hash function G: we use a combination of SHA-3 and Vision similar to [16]. We set $(n_{v_2}, m_{v_2}, c_{v_2}, r_{v_2}) = (12, 24, 2, 22)$.
- Symmetric-key encryption Π_{s}: We use AES-CTR with $\ell_{\mathrm{key}} = 256$ bits key (i.e., $K_{\mathrm{s}} = \{0, 1\}^{256}$). For simplicity, we assume the ciphertext length is identical to the plaintext length, i.e., $|\mathsf{Enc}_{\mathrm{s}}(\mathsf{k}, \mathsf{m})| = |\mathsf{m}|$.

From the above parameters, we have:

- output length of H_ℓ: $\ell_{\mathrm{hash}} = 32$ bytes,
- output length of H' and H'': $\ell_h = 64$ bytes.[10],
- output length of G: $\ell_g = 32$ bytes,
- ciphertext length of $\Pi_{\mathrm{k}}^{\mathrm{CM}}$: $|\mathsf{ct}_{\mathrm{k}}| = 96$ bytes,
- ciphetrext length of $\Pi_{\mathrm{p}}^{\mathrm{owCM}}$: $|\mathsf{ct}_{\mathrm{p}}^{\mathrm{ow}}| = 96$ bytes,
- verification key length of $\Pi_{\mathrm{sig}}^{\mathrm{Sig3}}$: $|\mathsf{vk}| = 165$ bytes,
- signature length of $\Pi_{\mathrm{sig}}^{\mathrm{Sig3}}$: $|\sigma| = 15{,}355$ bytes.

[9] The original CCMS conversion [16] uses Rescue that is suitable for MPCs over a large prime field since Lattice-based PKEs are working on such fields. In this work, we choose Vision that is suitable for MPC over characteristic two fields since hash input is a binary string in our constructions.

[10] For security, the output length of H' and H'' must be larger than $\lceil \binom{n_c}{t_c} \rceil = 457$ bits where $n_c = 3488$ and $t_c = 64$ in `kem/mceliece348864` Thus, we set $\ell_h = 512$ bits.

Table 1 Comparison of ciphertext overhead in the code-based (t, n)-threshold PKEs

	$\lvert\text{Ciphertext}\rvert - \lvert\text{plaintext}\rvert$ (bytes)
$\Pi_{\mathrm{t}}^{\mathrm{I}}$ (Sect. 4)	$n(\lvert\mathsf{ct}_{\mathrm{k}}\rvert + \ell_{\mathrm{hash}} + \ell_{\mathrm{key}}) + \lvert\sigma\rvert + \lvert\mathsf{vk}\rvert = 16{,}000$ B
$\Pi_{\mathrm{t}}^{\mathrm{II}}$ (Sect. 5)	$\binom{n}{t-1}(\lvert\mathsf{ct}_{\mathrm{p}}^{\mathrm{ow}}\rvert + \ell_{\mathrm{h}}) + \ell_{\mathrm{g}} = 512$ B
$\Pi_{\mathrm{t}}^{\mathrm{III}}$ (Sect. 6)	$\lvert\mathsf{ct}_{\mathrm{p}}^{\mathrm{ow}}\rvert + \ell_{\mathrm{h}} + \ell_{\mathrm{g}} = 192$ B

For concrete value, we consider the case $(t, n) = (2, 3)$ and 128-bit security

Table 2 Comparison of communication complexity during threshold decryption in the code-based (t, n)-threshold PKEs

	$\Pi_{\mathrm{p}}^{\mathrm{owCM}}.\mathsf{Dec}_{\mathrm{p}}$	MPC for hashing	
$\Pi_{\mathrm{t}}^{\mathrm{I}}$ (Sect. 4)	–	–	
$\Pi_{\mathrm{t}}^{\mathrm{II}}$ (Sect. 5)	–	$2 \cdot 7m_{v_1}\left(\max\left(\left\lceil\frac{\binom{n}{t-1}n_c}{n_{v_1}r_{v_1}}\right\rceil, 10\right) + \left\lceil\frac{\binom{n}{t-1}\ell_h}{n_{v_1}r_{v_1}}\right\rceil\right)$ $+7m_{v_2}\left(\max\left(\left\lceil\frac{\binom{n}{t-1}\ell_h}{n_{v_2}r_{v_2}}\right\rceil, 10\right) + \left\lceil\frac{\ell_g}{n_{v_2}r_{v_2}}\right\rceil\right)$	$= 18,816$
$\Pi_{\mathrm{t}}^{\mathrm{III}}$ (Sect. 6)	$O(t_c^3 + n_c t_c)$ $> 485,376$	$2 \cdot 7m_{v_1}\left(\max\left(\left\lceil\frac{n_c}{n_{v_1}r_{v_1}}\right\rceil, 10\right) + \left\lceil\frac{\ell_h}{n_{v_1}r_{v_1}}\right\rceil\right)$ $+7m_{v_2}\left(\max\left(\left\lceil\frac{\ell_h}{n_{v_2}r_{v_2}}\right\rceil, 10\right) + \left\lceil\frac{\ell_g}{n_{v_2}r_{v_2}}\right\rceil\right)$	$= 7728$

We give the complexity as the number of invocations of the multiplication protocol over $\mathbb{F}_{2^{12}}$. For the concrete values, we consider the case $(t, n) = (2, 3)$ and the 128-bit security parameters explained in the main body. "–" indicates the scheme does not perform the corresponding MPC

Table 1 shows the ciphertext overheads (the ciphertext length minus the plaintext length) of the $(2, 3)$-threshold PKEs. The ciphertext length of $\Pi_{\mathrm{t}}^{\mathrm{I}}$ comes from n IND-CCA PKE ciphertexts, one signature, and one verification key. Due to its large signature (about 15 kB), it is the largest overhead (about 16 kB) among the three threshold PKEs. The ciphertext overhead of $\Pi_{\mathrm{t}}^{\mathrm{II}}$ consists of $N = \binom{n}{t-1}$ OW-CPA PKE ciphertexts and one hash digest of G and one hash digest of H''. Note that the digest length of H'' is $N \cdot \ell_h$. Although its overhead depends on the number of parties, it is much smaller than that of $\Pi_{\mathrm{t}}^{\mathrm{I}}$ since it does not require signature schemes. The ciphertext length of $\Pi_{\mathrm{t}}^{\mathrm{III}}$ is derived from one OW-CPA PKE ciphertext, one digest of G, and one digest of H''. Thanks to the independence of the number of parties and the unnecessity of signatures, $\Pi_{\mathrm{t}}^{\mathrm{III}}$ has the smallest ciphertext overhead.

Table 2 shows the communication complexity of threshold decryption. We measure it by counting the number of invocations of the MPC for multiplication. The decryption protocol of $\Pi_{\mathrm{t}}^{\mathrm{I}}$ is non-interactive. So, parties need minimum communication costs. $\Pi_{\mathrm{t}}^{\mathrm{II}}$ can compute the decryption process of $\Pi_{\mathrm{p}}^{\mathrm{owCM}}$ and that of SKE locally. Therefore, the parties only need to communicate in the MPC of hashing for the validity check of ciphertexts. The MPC of Vision requires $7 \cdot m_v$ multiplications to absorb or extract $n_v \cdot r_v$ bits in each round. Note that the minimum absorption round is 10 in the parameters above. $\Pi_{\mathrm{t}}^{\mathrm{III}}$ performs, in addition to the MPC of hashing as in $\Pi_{\mathrm{t}}^{\mathrm{II}}$, the MPC for decryption of $\Pi_{\mathrm{p}}^{\mathrm{owCM}}$. In Table 2 we evaluate the cost dividing into two parts, one for decryption of $\Pi_{\mathrm{p}}^{\mathrm{owCM}}$, the other for hashing for verification. It shows that the MPC for decryption of $\Pi_{\mathrm{p}}^{\mathrm{owCM}}$ gives much more impact than that for hashing.

From the above discussion, we conclude that:

- Π_t^{I} achieves non-interactive threshold decryption but has much larger ciphertexts due to the impractical code-based signature scheme. So, this construction is not practical.
- Π_t^{II} has smaller ciphertext than Π_t^{I} by eliminating the use of a signature scheme. Also, Π_t^{II} achieves non-interactive decryption of OW-CPA secure PKE using parallel encryption. Although MPC for hashing needs to process longer input than Π_t^{III}, the input length would not impact so much. It means that Π_t^{II} is more efficient threshold PKE than Π_t^{III}. Overall, we can say that this construction has a well-balanced performance.
- Π_t^{III} has the worst efficiency for threshold decryption but the smallest ciphertext length. It also has the smallest public-key length, unlike the other two constructions that need three times longer public keys even in the (2, 3)-setting. Therefore, this construction is the best solution, if decryption parties can devote ample communication cost.

8 Conclusion

In this paper, we have proposed three code-based threshold PKE schemes from Classic McEliece. These three schemes have a trade-off between ciphertext lengths and communication costs of threshold decryption. The first one has a very long ciphertext, but it allows decrypting ciphertexts non-interactively, i.e., without MPCs. The second one has about 97% shorter ciphertext than the first one, and requires only MPCs for computing hash functions. The third one has the shortest ciphertext but requires heavy MPCs to decrypt a ciphertext. Among the three schemes, the second one has a good balance between ciphertext size and communication complexity.

The first scheme is impractical at this moment because of lengthy ciphertexts. 96% of the ciphertext length comes from the signature-related data appended to guarantee CCA security. Recently, code-based signature schemes have been actively studied, and new schemes have been proposed one after another [8, 13, 25]. So, once highly efficient code-based signature schemes with strong one-time EUF-CMA security are developed, the first scheme could become fascinating.

We note that the approach we take to construct the second scheme is applicable not only to code-based PKEs but also to any PKEs with some properties. So, we immediately obtain other post-quantum instantiations, once we prepare a PKE with the properties. Unlike the Dodis–Katz conversion, which uses a signature scheme (with *strong* one-time EUF-CMA security), our conversion does not utilize signature schemes. Therefore, our approach allows us to derive a threshold PKE scheme whose ciphertext is only constant times longer than the standard PKE, regardless of whether post-quantum signatures are large.

If we want more short ciphertexts, we can take the third approach but need to design an efficient MPC for decryption. It is an interesting open problem to construct other post-quantum threshold PKEs which utilize efficient threshold decryption.

Data availability Data sharing is not applicable to this article as no datasets were generated or analysed during the study.

References

1. Adida B.: Helios: web-based open-audit voting. In: van Oorschot P.C. (ed.) USENIX Security 2008, pp. 335–348. USENIX Association (2008).
2. Albrecht M.R., Bernstein D.J., Chou T., Cid C., Gilcher J., Lange T., Maram V., von Maurich I., Misoczki R., Niederhagen R., Paterson K.G., Persichetti E., Peters C., Schwabe P., Sendrier N., Szefer J., Tjhai C.J., Tomlinson M., Wang W.: Classic McEliece. Technical report. National Institute of Standards and Technology (2022). https://csrc.nist.gov/projects/post-quantum-cryptography/round-4-submissions.
3. Aly A., Ashur T., Ben-Sasson E., Dhooghe S., Szepieniec A.: Design of symmetric-key primitives for advanced cryptographic protocols. IACR Trans. Symm. Cryptol. **2020**(3), 1–45 (2020). https://doi.org/10.13154/tosc.v2020.i3.1-45.
4. An J.H., Dodis Y., Rabin T.: On the security of joint signature and encryption. In: Knudsen L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 83–107. Springer (2002). https://doi.org/10.1007/3-540-46035-7_6.
5. Bar-Ilan J., Beaver D.: Non-cryptographic fault-tolerant computing in constant number of rounds of interaction. In: Rudnicki P. (ed.) 8th ACM PODC, pp. 201–209. ACM (1989). https://doi.org/10.1145/72981.72995.
6. Bendlin R., Damgård I.: Threshold decryption and zero-knowledge proofs for lattice-based cryptosystems. In: Micciancio D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 201–218. Springer (2010). https://doi.org/10.1007/978-3-642-11799-2_13.
7. Bernstein D.J., Persichetti E.: Towards KEM unification. Cryptology ePrint Archive, Report 2018/526. https://eprint.iacr.org/2018/526 (2018).
8. Bidoux L., Gaborit P., Kulkarni M., Mateu V.: Code-based signatures from new proofs of knowledge for the syndrome decoding problem. Des. Codes Cryptogr. **91**(2), 497–544 (2023). https://doi.org/10.1007/s10623-022-01114-3.
9. Bindel N., Hamburg M., Hövelmanns K., Hülsing A., Persichetti E.: Tighter proofs of CCA security in the quantum random oracle model. In: Hofheinz D., Rosen A. (eds.) TCC 2019, Part II. LNCS, vol. 11892, pp. 61–90. Springer (2019). https://doi.org/10.1007/978-3-030-36033-7_3.
10. Brandao L.T.A.N., Mouha N., Vassilev A.: Threshold schemes for cryptographic primitives. NIST Interagency/Internal Report (NISTIR). National Institute of Standards and Technology, Gaithersburg (2019). https://doi.org/10.6028/NIST.IR.8214.
11. Burkhart M., Strasser M., Many D., Dimitropoulos X.A.: SEPIA: privacy-preserving aggregation of multi-domain network events and statistics. In: USENIX Security 2010, pp. 223–240. USENIX Association (2010).
12. Canetti R., Goldwasser S.: An efficient threshold public key cryptosystem secure against adaptive chosen ciphertext attack. In: Stern J. (ed.) EUROCRYPT'99. LNCS, vol. 1592, pp. 90–106. Springer (1999). https://doi.org/10.1007/3-540-48910-X_7.
13. Carozza E., Couteau G., Joux A.: Short signatures from regular syndrome decoding in the head. In: Hazay C., Stam M. (eds.) EUROCRYPT 2023, Part V. LNCS, vol. 14008, pp. 532–563. Springer (2023). https://doi.org/10.1007/978-3-031-30589-4_19.
14. Cayrel P.-L., Véron P., El Yousfi Alaoui S.M.: A zero-knowledge identification scheme based on the q-ary syndrome decoding problem. In: Biryukov A., Gong G., Stinson D.R. (eds.) SAC 2010. LNCS, vol. 6544, pp. 171–186. Springer (2011). https://doi.org/10.1007/978-3-642-19574-7_12.
15. Clarkson M.R., Chong S., Myers A.C.: Civitas: toward a secure voting system. In: 2008 IEEE Symposium on Security and Privacy (SP 2008), pp. 354–368 (2008). https://doi.org/10.1109/SP.2008.32.
16. Cong K., Cozzo D., Maram V., Smart N.P.: Gladius: LWR based efficient hybrid public key encryption with distributed decryption. In: Tibouchi M., Wang H. (eds.) ASIACRYPT 2021, Part IV. LNCS, vol. 13093, pp. 125–155. Springer (2021). https://doi.org/10.1007/978-3-030-92068-5_5.
17. Courtois N., Finiasz M., Sendrier N.: How to achieve a McEliece-based digital signature scheme. In: Boyd C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 157–174. Springer (2001). https://doi.org/10.1007/3-540-45682-1_10.
18. Cramer R., Gennaro R., Schoenmakers B.: A secure and optimally efficient multi-authority election scheme. In: Fumy W. (ed.) EUROCRYPT'97. LNCS, vol. 1233, pp. 103–118. Springer (1997). https://doi.org/10.1007/3-540-69053-0_9.
19. Cramer R., Shoup V.: Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. SIAM J. Comput. **33**(1), 167–226 (2003).
20. Damgård I., Fitzi M., Kiltz E., Nielsen J.B., Toft T.: Unconditionally secure constant-rounds multi-party computation for equality, comparison, bits and exponentiation. In: Halevi S., Rabin T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 285–304. Springer (2006). https://doi.org/10.1007/11681878_15.

21. Damgård I., Jurik M.: A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. In: Kim K. (ed.) PKC 2001. LNCS, vol. 1992, pp. 119–136. Springer (2001). https://doi.org/10.1007/3-540-44586-2_9.
22. Debris-Alazard T., Sendrier N., Tillich J.-P.: Wave: a new family of trapdoor one-way preimage sampleable functions based on codes. In: Galbraith S.D., Moriai S. (eds.) ASIACRYPT 2019, Part I. LNCS, vol. 11921, pp. 21–51. Springer (2019). https://doi.org/10.1007/978-3-030-34578-5_2.
23. Desmedt Y., Frankel Y.: Threshold cryptosystems. In: Brassard G. (ed.) CRYPTO'89. LNCS, vol. 435, pp. 307–315. Springer (1990). https://doi.org/10.1007/0-387-34805-0_28.
24. Dodis Y., Katz J.: Chosen-ciphertext security of multiple encryption. In: Kilian J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 188–209. Springer (2005). https://doi.org/10.1007/978-3-540-30576-7_11.
25. Feneuil T., Joux A., Rivain M.: Syndrome decoding in the head: shorter signatures from zero-knowledge proofs. In: Dodis Y., Shrimpton T. (eds.) CRYPTO 2022, Part II. LNCS, vol. 13508, pp. 541–572. Springer (2022). https://doi.org/10.1007/978-3-031-15979-4_19.
26. Feneuil T., Joux A., Rivain M.: Shared permutation for syndrome decoding: new zero-knowledge protocol and code-based signature. Des. Codes Cryptogr. **91**(2), 563–608 (2023). https://doi.org/10.1007/s10623-022-01116-1.
27. Fiat A., Shamir A.: How to prove yourself: practical solutions to identification and signature problems. In: Odlyzko A.M. (ed.) CRYPTO'86. LNCS, vol. 263, pp. 186–194. Springer (1987). https://doi.org/10.1007/3-540-47721-7_12.
28. Frankel Y., Desmedt Y.: Parallel reliable threshold multisignature. Technical report TR-92-04-02. University of Wisconsin-Milwaukee (1992).
29. Fujisaki E., Okamoto T.: Secure integration of asymmetric and symmetric encryption schemes. J. Cryptol. **26**(1), 80–101 (2013). https://doi.org/10.1007/s00145-011-9114-1.
30. Goppa V.D.: A new class of linear correcting codes. Problemy Peredachi Informatsii **6**(3), 24–30 (1970).
31. Gueron S., Persichetti E., Santini P.: Designing a practical code-based signature scheme from zero-knowledge proofs with trusted setup. Cryptography (2022). https://doi.org/10.3390/cryptography6010005.
32. Ishida N., Matsuo S., Ogata W.: Efficient divisible voting scheme. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **88**(1), 230–238 (2005). https://doi.org/10.1093/ietfec/e88-a.1.230.
33. Kokoris-Kogias E., Alp E.C., Gasser L., Jovanovic P., Syta E., Ford B.: Calypso: private data management for decentralized ledgers. Proc. VLDB Endow. **14**(4), 586–599 (2020). https://doi.org/10.14778/3436905.3436917.
34. Kraitsberg M., Lindell Y., Osheter V., Smart N.P., Talibi Alaoui Y.: Adding distributed decryption and key generation to a ring-LWE based CCA encryption scheme. In: Jang-Jaccard J., Guo F. (eds.) ACISP 19. LNCS, vol. 11547, pp. 192–210. Springer (2019). https://doi.org/10.1007/978-3-030-21548-4_11.
35. Krawczyk H.: Secret sharing made short. In: Stinson D.R. (ed.) CRYPTO'93. LNCS, vol. 773, pp. 136–146. Springer (1994). https://doi.org/10.1007/3-540-48329-2_12.
36. Mohassel P., Franklin M.: Efficient polynomial operations in the shared-coefficients setting. In: Yung M., Dodis Y., Kiayias A., Malkin T. (eds.) PKC 2006. LNCS, vol. 3958, pp. 44–57. Springer (2006). https://doi.org/10.1007/11745853_4.
37. National Institute of Standards and Technology: Multi-party Threshold Cryptography. https://csrc.nist.gov/projects/Threshold-Cryptography. Accessed 3 Jan 2023.
38. National Institute of Standards and Technology: Post-quantum Cryptography. https://csrc.nist.gov/projects/post-quantum-cryptography. Accessed 3 Jan 2023.
39. Nikova S., Rechberger C., Rijmen V.: Threshold implementations against side-channel attacks and glitches. In: Ning P., Qing S., Li N. (eds.) ICICS 06. LNCS, vol. 4307, pp. 529–545. Springer (2006).
40. Patterson N.: The algebraic decoding of Goppa codes. IEEE Trans. Inf. Theory (1975). https://doi.org/10.1109/TIT.1975.1055350.
41. Shamir A.: How to share a secret. Commun. Assoc. Comput. Mach. **22**(11), 612–613 (1979).
42. Shor P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J. Comput. **26**(5), 1484–1509 (1997). https://doi.org/10.1137/S0097539795293172.
43. Shoup V., Gennaro R.: Securing threshold cryptosystems against chosen ciphertext attack. In: Nyberg K. (ed.) EUROCRYPT'98. LNCS, vol. 1403, pp. 1–16. Springer (1998). https://doi.org/10.1007/BFb0054113.
44. Shoup V.: A proposal for an ISO standard for public key encryption. Cryptology ePrint Archive, Report 2001/112. https://eprint.iacr.org/2001/112 (2001).
45. Stern J.: A new identification scheme based on syndrome decoding. In: Stinson D.R. (ed.) CRYPTO'93. LNCS, vol. 773, pp. 13–21. Springer (1994). https://doi.org/10.1007/3-540-48329-2_2.
46. Véron P.: Improved identification schemes based on error-correcting codes. Appl. Algebra Eng. Commun. Comput. **8**(1), 57–69 (1997). https://doi.org/10.1007/s002000050053.

47. von zur Gathen J., Gerhard J.: Modern Computer Algebra, 3rd edn. Cambridge University Press, Cambridge (2013). https://doi.org/10.1017/CBO9781139856065.
48. Yakira D., Asayag A., Cohen G., Grayevsky I., Leshkowitz M., Rottenstreich O., Tamari R.: Helix: a fair blockchain consensus protocol resistant to ordering manipulation. IEEE Trans. Netw. Serv. Manag. **18**(2), 1584–1597 (2021). https://doi.org/10.1109/TNSM.2021.3052038.