Chosen Ciphertext Secure Pseudorandom Codes in the Standard Model

Nico Döttling[1], Antoine Joux[1], Venkata Koppula[2],
Mahesh Sreekumar Rajasree[1], and Hendrik Waldner[1]

[1] CISPA Helmholtz Center for Information Security, Saarbrücken, Germany
{doettling, joux, hendrik.waldner}@cispa.de, srmahesh1994@gmail.com
[2] Indian Institute of Technology Delhi, New Delhi, India
kvenkata@iitd.ac.in

Abstract. Pseudorandom codes (PRCs), recently proposed by Christ and Gunn (CRYPTO'24), are encryption schemes that have pseudorandom ciphertexts and a decryption algorithm which is resilient against a bounded number of Hamming errors. This notion provides a significant strengthening over standard PKE and has exciting applications in, e.g., watermarking LLMs. The recent work of Alrabiah et al. (STOC'25) initiated the study of CCA-secure public-key PRCs, where the adversary is additionally given access to a decoding oracle. In terms of realizations, they provide one that can be proven secure in the random oracle model. Constructing CCA-secure public-key PRCs in the standard model remained an open problem.

In this work, we resolve this problem and provide the first construction of CCA-secure public-key PRCs in the standard model. Our construction achieves a constant rate and can decode from a constant fraction of adversarial errors. In fact, our construction is a general blueprint that can be instantiated from a broad range of standard cryptographic assumptions.

As an additional contribution, we construct a strong adaptively robust public-key pseudorandom code with conjectured sub-exponential security based on a new family of assumptions we call Noisy McEliece. In a nutshell, these assumptions mask a scrambled generator matrix from an efficiently decodable inner-code family with sparse Bernoulli noise; this additional error is meant to obscure the algebraic structure targeted by known attacks and thereby permits candidate instantiations from broader classes of codes.

1 Introduction

1.1 Pseudorandom Codes

Error-correcting codes and pseudorandomness are two of the most fundamental notions in theoretical computer science and cryptography. Error-correcting codes enable reliable communication over noisy channels, while pseudorandom generators (PRGs) are indispensable for derandomization and cryptographic security. While error-correction requires redundancy and hence structure, pseudorandomness makes any structure undetectable.

The notion of pseudorandom codes (PRCs), recently proposed by Christ and Gunn [CG24], connects these seemingly contradictory notions. In a nutshell, a PRC is a family of keyed error-correcting codes whose codewords are computationally indistinguishable from uniformly random ones to an outside observer. But, a legitimate receiver who is in possession of the secret key can efficiently decode the codeword even in the presence of a large fraction of Hamming errors. While pseudorandom codes were originally motivated by the problem of watermarking large language models (LLMs), they have emerged as an exciting cryptographic notion in their own right.

However, constructing pseudorandom codes has turned out to be a highly non-trivial task. Current constructions [CG24, GG24, GM24, AAC$^+$25, CGG$^+$25] are based on hardness assumptions relating to coding theory, and there is evidence that even constructing just the weakest private-key notions cannot be achieved from *generic* hardness assumptions [DMR25, GGW25]. In particular, [GGW25] argues that even the most powerful tools available in cryptography such as (indistinguishability) obfuscation do not help when attempting to construct pseudorandom codes with generic recipes.

1.2 Adaptive Robustness and CCA Security

Adaptive Robustness: Early PRC constructions addressed errors introduced obliviously by a memoryless channel. Subsequent works [CHS25, AAC$^+$25] argued that, in realistic settings, robustness must be defined against *adaptive* adversaries that have access to an encoding oracle (or to the public key in the public-key setting). Such an adversary must output a message m, randomness r, and a c such that encoding m with randomness r yields a codeword close to c, yet decoding c produces a different message. In this work, we rely on this stronger notion of robustness.

CCA security: The standard security notion for pseudorandom codes [CG24] mirrors the basic *chosen plaintext* security notion (CPA) for encryption schemes with pseudorandom ciphertexts: the adversary is given access to an oracle $\mathcal{O}_{Enc}$ that produces strings c_i (as well as potentially the PRC public key) and must decide whether the strings produced by the oracle are well-formed codes of the PRC or just uniformly random strings. Several real-world attacks [JSV24, PHZS24] exploit access to both the encoding and decoding oracles in order to remove/counterfeit watermarks. Alrabiah et al. [AAC$^+$25] introduced stronger security notions in the private key and public key setting, where the adversary also has access to the decoding oracle. In the private key setting, they introduced the notion of *ideal pseudorandom codes* for pseudorandom codes. In this notion, the adversary is additionally given access to a *decoding* oracle $\mathcal{O}_{Dec}$ which provides the adversary access to the decoding algorithm. Implementing this oracle for the case that $\mathcal{O}_{Enc}$ actually implements the encoding algorithm is easy. However, if $\mathcal{O}_{Enc}$ produces uniformly random strings, a more refined argument is required. Specifically, in this case $\mathcal{O}_{Enc}$ and $\mathcal{O}_{Dec}$ must share a joint state, and $\mathcal{O}_{Dec}$ needs to handle strings produced by $\mathcal{O}_{Enc}$ differently from codewords produced by the adversary.
In the public key setting, which is the focus of this work, [AAC$^+$25] proposed a simple *chosen ciphertext* security notion (CCA) and provided a construction in the random oracle model [BR93]. Specifically, in this security notion the adversary is given the public key, as well as access to a decoding oracle $\mathcal{O}_{Dec}$ which implements the decoding algorithm. At some point, the adversary asks for a challenge and is either given a well-formed codeword, or a uniformly random string. As in the standard CCA experiment, access to the decoding oracle is now restricted. While, at this point in the CCA experiment for public key encryption, the decryption oracle accepts every query *except* the challenge ciphertext, this formulation is insufficient for pseudorandom codes. Specifically, the adversary can always introduce a small amount of errors to the challenge and obtain a different string. Hence, in the PRC-setting the post-challenge decoding oracle must be restricted in a more stringent way. Roughly speaking, the post-challenge decoding oracle now summarily rejects any input within (say) the decoding-radius of the challenge.

1.3 Constructing CCA-Secure Pseudorandom Codes

However, in light of mounting impossibility results for PRCs [DMR25, GGW25], and given the growing body of random oracle uninstantiability results [Nie02, GK03, BBP04, LN09, Zha22, BDD22], the very existence of the notion of CCA-secure PRC rests on uncertain grounds. This naturally raises the following question:

Can CCA-secure pseudorandom codes be realized in the standard model under standard assumptions?

1.4 Our Results

In this work, we present the first construction of public-key pseudorandom codes that achieves indistinguishability against chosen-ciphertext attacks in the standard model. We also present a new candidate construction of an adaptively robust PRC.

New Candidates for Robust PRCs. Existing constructions of robust public-key PRCs need to assume sub-exponential hardness of LPN [AAC$^+$25], while only providing quasi-polynomial security. Furthermore, constructions attaining sub-exponential security are known only in the secret-key setting [CGG$^+$25]. As an additional contribution, we propose a new family of assumptions which we call *Noisy McEliece* assumptions.

Recall that the decisional McEliece assumption [McE78] conjectures that the generator matrices of *scrambled Goppa codes* are pseudorandom. Specifically, if $\mathbf{G} \in \mathbb{F}_2^{m \times n}$ is the generator matrix of a binary Goppa code, $\mathbf{\Pi} \in S_m$ is a random permutation matrix, and $\mathbf{T} \in \mathsf{GL}_n(\mathbb{F}_2)$ is a random invertible matrix, then we set $\mathbf{A} = \mathbf{\Pi} \cdot \mathbf{G} \cdot \mathbf{T}$. Under the decisional McEliece assumption this matrix $\mathbf{A}$ is pseudorandom. Notoriously, virtually all attempts to replace binary Goppa codes in this construction with other efficiently decodable linear codes have led to insecure constructions [SS92, vT94, CC98, GOT12, CGGU$^+$14, COTGU15, Mce26]. These attacks show that the choice of the code family is central. In our Noisy McEliece assumptions, one fixes a specified code family and masks the scrambled generator matrix by Bernoulli noise, that is, we consider matrices of the form $\mathbf{A} = \mathbf{\Pi} \cdot \mathbf{G} \cdot \mathbf{T} + \mathbf{E}$, where $\mathbf{E} \leftarrow \mathsf{Ber}_\rho^{m \times n}$ is chosen from a $\rho = O(1/\sqrt{n})$-noise Bernoulli distribution. The hope is that the noise disrupts the exact algebraic invariants exploited by existing attacks, but this depends on the chosen code family. In Section 4 we discuss candidate code families, insecure regimes, and the need for further cryptanalysis, and provide a construction of an adaptively robust pseudorandom code with conjectured sub-exponential security from this assumption.

Public-Key Pseudorandom Codes with CCA Security. In Section 5, we describe a generic transformation for constructing an IND-CCA-secure PRC scheme using a primitive called *hinting pseudorandom generator (HPRG)*[KW19].

Theorem 1 (Informal). *Assuming the existence of an IND-CPA-secure PRC scheme with adaptive robustness, and a secure hinting PRG, there exists an IND-CCA-secure PRC scheme.*

Instantiations. Since every PRC is a PKE scheme, the required building blocks can be instantiated using several well-studied hardness assumptions:

- **Adaptively Robust PRC:** Under sub-exponential Learning Parity with noise (exLPN) [AAC$^+$25], or our noisy McEliece assumption.
- **Hinting PRGs:** Constructions are known from Learning with Errors (LWE) [KW19], Computational Diffie–Hellman (CDH) [KW19], Decisional Diffie–Hellman (DDH) [AP22].

These instantiations yield the following corollary.

Corollary 1. *Assuming the hardness of $\mathcal{X}$ and $\mathcal{Y}$, where $\mathcal{X} \in \{\mathsf{LWE}, \mathsf{DDH}, \mathsf{CDH}\}$ and $\mathcal{Y} \in \{$noisy McEliece, exLPN$\}$, there exists an IND-CCA-secure PRC scheme.*

Remark 1 (Instantiating our results). The above construction relies on a specialized form of hinting PRG, which we refer to as an *entropic hinting PRG*. This is essentially a hinting PRG that remains secure even when its seed is not sampled uniformly at random, but instead drawn from a distribution with sufficiently high entropy. Notably, most of the known constructions of hinting PRGs employ the *leftover hash lemma* in their security analysis, thereby ensuring the entropic property. For further details, see Section 3.5.
One exception to the above is the Linear Hidden Shift (LHS)-based construction of [AP22]. In the proof of this construction, the pseudorandomness of the output is directly argued by relying on the LHS assumption. While this is out of scope of our current work, we believe that the same proof strategy as in [AP22] gives rise to entropic hinting PRGs, if one is willing to rely on a high entropy version of the LHS assumption. If such an assumption is plausible, we would obtain an additional instantiation of the result above.

1.5 Related Works

Christ and Gunn [CG24] introduced the notion of PRCs and gave constructions in both, the secret-key and public-key settings. The subsequent work [AAC$^+$25] strengthened the security model by allowing the adversary to have access to the decoding algorithm, introducing the corresponding CCA notion. More recently, [CGG$^+$25] constructed secret-key PRCs that, for the first time, simultaneously achieve plausible subexponential pseudorandomness security, robustness to worst-case edit errors over a binary alphabet, and resilience even against computationally unbounded adversaries equipped with the decoding key. Achieving this combination of

properties requires overcoming several barriers, as prior constructions neither attained subexponential security nor provided strong resilience guarantees. Zero-bit PRCs[3] were constructed in [GM24] under the assumption of local weak pseudorandom functions, which in turn follow from the hardness of learning $\log(n)$-juntas over the uniform distribution on $\{0,1\}^n$. Additional constructions of pseudorandom codes were explored in [GG24]. In particular, they construct public-key PRCs based on the planted hyperloop assumption [BKR23] and pseudorandom generators, and also obtain statistically secure PRCs against space-bounded adversaries. On the contrary, two independent works [GGW25, DMR25] establish strong impossibility results for black-box constructions of pseudorandom codes from a broad class of strong cryptographic primitives including virtual black-box obfuscation.

Next, we give a brief overview of existing IND-CPA to IND-CCA transformations. None of these transformations consider efficient error correction, which makes them unsuitable for amplifying the security of pseudorandom codes directly. We categorize the existing transformations in the following way:

1. **Proof-Carrying Ciphertexts**: This is the classic paradigm for constructing CCA-secure PKE. In this paradigm, multiple ciphertexts of a CPA secure encryption scheme, generated under different public keys, are *glued together* using a NIZK proof of well-formedness. Such proofs can be publicly verifiable (e.g. [NY90, DDN91, Sah99]) or designated verifier (e.g. [CS98, CS02, QRW19, LQR$^+$19]). Evidently, using publicly verifiable NIZK proofs is incompatible with the goal of constructing CCA encryption with pseudorandom ciphertexts. By contrast, schemes based on designated verifier proofs can indeed yield CCA-secure encryption schemes with pseudorandom ciphertexts, as exemplified by the well-known Cramer-Shoup encryption scheme [CS98, BBDP01].
2. **Tag/Label-based Encryption and the CHK-Transform**: An alternative standard-model approach to constructing CCA-secure PKE follows the CHK-paradigm [CHK03, BCHK07, BK05] to transform an identity-based encryption (IBE) scheme or, more generally, a tag-based encryption (TBE) scheme [Kil06] by embedding a one-time signature verification key into the identity/tag, and by including a signature of the IBE/TBE ciphertext into the CCA ciphertext. This blueprint can be instantiated from a broad range of assumptions, including pairings [CHK03], lossy trapdoor functions [PW08], lattice assumptions [MP12], coding assumptions [DMN12, KMP14], hinting PRGs [KW19] and injective trapdoor functions [HKW20]. This paradigm does not typically yield CCA-encryption with pseudorandom ciphertexts, since the ciphertext includes both, a signature and a corresponding verification key, which obey a publicly testable relation. One exception here is the transformation of Boneh and Katz [BK05], which has been shown in [Xag22]. In this work, the author shows that the Boneh-Katz transformation achieves pseudorandom ciphertexts if all of the underlying primitives are pseudorandom as well.
3. **CCA-security via KEM**: In [KMT19], besides multiple other results, the authors show how to realize an IND-CCA secure key encapsulation mechanism (KEM) from an IND-CPA secure KEM as well as a one-time key-dependent-message (KDM) secure SKE scheme. Relying on the IND-CCA secure KEM, an IND-CCA secure PKE scheme can then be obtained using a hybrid encryption approach. To obtain the IND-CPA secure KEM, the authors follow the framework of Koppula and Waters [KW19], besides also mentioning other possible instantiations [BHHO08, ACPS09, BG10, BLSV18, DGHM18]. Additionally, the authors also show how their IND-CCA transformation can be adapted to obtain IND-$CCA secure KEMs[4]. To achieve this result, the presented compiler is instantiated using an IND-$CPA secure KEM, instead of an IND-CPA secure one, as well as a one-time randomness-recoverable KDM secure SKE with pseudorandom ciphertexts. Achieving IND-$CPA secure KEM, as well as amplifying an existing one-time KDM secure SKE by adding randomness recoverability and pseudorandom ciphertexts, can be easily achieved. On a more technical level, in this transformation, the authors first realize an adaptively secure TDF which directly implies the desired IND-$CCA secure KEM. Given that the authors of [KMT19] generalized their transformation for adaptively secure TDFs, it is also possible to use the work of [KM25], in which the authors construct an adaptively secure TDF with pseudorandom strings based on a standard TDF and a PRG, to obtain IND-$CCA secure KEMs.

[3] A zero-bit PRC can encode only a single message $\mathsf{msg} = 0$.

[4] An IND-$CCA secure KEM can be turned into an IND-$CCA secure PKE using the same hybrid encryption approach as for the IND-CCA case.

4. **Constructions in the Random Oracle Model**: The random oracle model (ROM) [BR93] greatly simplifies the construction of CCA-secure encryption schemes as it facilitates the simulation of the decryption oracle. Specifically, the random oracle enables extraction of the ciphertext randomness of adversarially generated ciphertexts, hence decryption queries can be answered without access to the secret key. Constructing CCA-secure encryption schemes in the ROM is typically practically motivated, as the resulting CCA-secure encryption schemes usually achieve essentially the same efficiency as the underlying CPA-secure encryption schemes. In fact, the random oracle model was formalized precisely for this purpose [BR93]. The Fujisaki-Okamoto transform [FO99] allows to transform any CPA-secure PKE into a CCA-secure encryption scheme without changing the ciphertext structure. Hence, in particular, it enables the construction of CCA-secure encryption schemes with pseudorandom ciphertexts from any CPA-secure encryption scheme with this property. On a conceptual level, while constructions in the ROM yield practically efficient schemes, they do not provide deep foundational insights.

As already mentioned in Section 1.4, in this work, we rely on the hinting PRG framework of Koppula-Waters [KW19]. We leave it as an interesting open problem if any of the other mentioned CCA transformations can also be applied to handle efficient error correction and obtain $\mathsf{IND\text{-}CCA}$ secure PRCs.

2 Technical Overview

In this section, we give a high-level overview of our construction. We begin by recalling the Koppula-Waters (KW) [KW19] construction of an $\mathsf{IND\text{-}CCA\text{-}2}$ scheme.

2.1 The KW construction.

The KW construction gave a template for upgrading any $\mathsf{IND\text{-}CPA}$ PKE scheme to a CCA secure one, using a special kind of pseudorandom generators called *hinting PRGs*. A hinting PRG is a deterministic function $\mathsf{HPRG} : \{0,1\}^\lambda \to \{0,1\}^{\ell+\lambda^2}$ (where ℓ is polynomial in λ) with the following security property: for any PPT adversary, the following distribution looks uniform:

- sample seed $s \leftarrow \{0,1\}^\lambda$ and compute $\mathsf{HPRG}(s) = (r_0, r_1, \ldots, r_\lambda)$. Here $r_0 \in \{0,1\}^\ell$ and each $r_i \in \{0,1\}^\lambda$.
- For each i, set $y_{i,s_i} = r_i$, and sample $y_{i,\overline{s_i}} \leftarrow \{0,1\}^\lambda$.
- Output $\left(r_0, (y_{i,b})_{i,b}\right)$.

Additionally, the KW construction uses *perfectly binding tag-based equivocal bit commitments*. In a tag-based commitment scheme, we have a setup algorithm that outputs the public parameters. The commitment algorithm Com takes as input the public parameters, the message bit and the tag, and produces a commitment string. For perfect binding, we require that for any public parameters pp, any tag, there does not exist randomness v_0, v_1 such that $\mathsf{Com}(\mathsf{pp}, 0, \mathsf{tag}; v_0) = \mathsf{Com}(\mathsf{pp}, 1, \mathsf{tag}; v_1)$. The equivocal property states that it is possible to generate 'fake' public parameters pp^* together with randomness v_0^*, v_1^* for a challenge tag^* such that pp^* is computationally indistinguishable from the regular public parameters, but $\mathsf{Com}(\mathsf{pp}^*, 0, \mathsf{tag}^*; v_0^*) = \mathsf{Com}(\mathsf{pp}^*, 1, \mathsf{tag}^*; v_1^*)$. Almost perfectly binding tag-based equivocal bit commitment schemes can be constructed using any (regular) PRG [Nao90, KW19].

The KW transformation also requires the underlying PKE scheme to satisfy the following two properties:

- *almost perfect correctness* - for almost all public keys pk, for all messages m and randomness r, $\mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, m; r)) = m$; and
- *decryption using randomness* - ciphertexts can be decrypted given the encryption randomness.

Both these properties can be achieved for free. Dwork, Naor and Reingold [DNR04a] showed that any $\mathsf{IND\text{-}CPA}$ PKE scheme can be transformed into another PKE scheme that satisfies almost perfect correctness. Likewise, any PKE scheme can be modified to also support decryption using randomness [KW19]. Now, we describe the setup, encryption and decryption algorithms of the KW scheme:

Key generation. The setup algorithm generates key pairs $(\mathsf{pk}_{i,b}, \mathsf{sk}_{i,b})$ for all $i \in [\lambda]$, $b \in \{0,1\}$, where λ is the seed length of HPRG. The public key is the collection $\left(\mathsf{pk}_{i,b}\right)_{i,b}$, while the secret key consists of $\left(\mathsf{sk}_{i,0}\right)_i$.[5]

Encryption. To encrypt a message m, the sender samples a random seed $s \in \{0,1\}^\lambda$ and computes $(r_0, r_1, \ldots, r_\lambda) = \mathsf{HPRG}(s)$. Next, it samples a one-time signing/verification key $(\mathsf{ds.sk}, \mathsf{ds.vk})$ and randomness $\left(v_i\right)_i$ for the commitment scheme. The ciphertext consists of two parts:

- The payload $c = m \oplus r_0$.
- For each $i \in [\lambda]$, it computes a commitment to the bit s_i with tag $\mathsf{ds.vk}$ and randomness v_i (that is, $\mathsf{com}_i \leftarrow \mathsf{Com}(s_i, \mathsf{ds.vk}; v_i)$). Then, it sets

$$c_{i,s_i} = \mathsf{Enc}\left(\mathsf{pk}_{i,s_i}, 1\|v_i; \tilde{r}_i\right), \qquad c_{i,\overline{s_i}} \leftarrow \mathsf{Enc}\left(\mathsf{pk}_{i,\overline{s_i}}, 0^\lambda\right)$$

Finally, the encryption algorithm signs $\mathsf{ds}.x = \left(c, \left(c_{i,b}, \mathsf{com}_i\right)_{i,b}\right)$ using $\mathsf{ds.sk}$, resulting in the signature $\mathsf{ds}.\sigma$. The full ciphertext is $(\mathsf{ds.vk}, \mathsf{ds}.x, \mathsf{ds}.\sigma)$.

Decryption. First, the decryption algorithm verifies the signature using the verification key $\mathsf{ds.vk}$. If the signature verifies, the decryption algorithm recovers a candidate seed s, performs consistency checks using s, and finally recovers the message from the payload. More formally,

1. *Recovering the seed.* For each i, it uses $\mathsf{sk}_{i,0}$ to decrypt $c_{i,0}$. If the result is of the form $1\|v_i$ and v_i is a valid opening of com_i to 0 (with respect to tag $\mathsf{ds.vk}$), it sets $s_i = 0$. Otherwise, it sets $s_i = 1$. This yields a candidate seed s.
2. *Verification and message recovery.* Using s, it recomputes $(r_0, (r_i)_i) = \mathsf{HPRG}(s)$. For each $i \in [\lambda]$,
   - it checks that decrypting c_{i,s_i} with randomness r_i [6] yields a message of the form $1\|w_i$, and that w_i is a valid opening of com_i for bit s_i (with respect to $\mathsf{ds.vk}$)
   - it checks that re-encrypting $1\|w_i$ using randomness r_i yields c_{i,s_i}.

   If all checks pass, it outputs $m = c \oplus r_0$; otherwise, it outputs $\perp$.

The KW scheme, like many other CCA secure encryption schemes, uses the *secret hiding principle* [FS90], where we have multiple secret keys for the same public key. In the security proof, we switch from using one secret key to another. This is a key technical step in the security proof of KW. Note that in the above described construction, for any $\mathfrak{p} \in \{0,1\}^n$, we could have used the secret keys $\left(\mathsf{sk}_{i,\mathfrak{p}_i}\right)_i$ for decryption, and correctness would still hold. KW showed that even for maliciously crafted ciphertexts, decryption of ct using $\left(\mathsf{sk}_{i,0}\right)_i$ is equal to the decryption of ct using $\left(\mathsf{sk}_{i,\mathfrak{p}_i}\right)_i$. Below, we briefly discuss why this property holds (assuming almost-perfect correctness of PKE and almost-perfect binding of the commitment scheme).
Let $\mathsf{Dec}^\dagger$ denote the decryption using $\left(\mathsf{sk}_{i,\mathfrak{p}_i}\right)_{i\in[\lambda]}$. Fix any ciphertext ct, and let s (resp. $s^\dagger$) denote the candidate seed found in the first step of Dec (resp. $\mathsf{Dec}^\dagger$). Clearly, $s \neq s^\dagger$, otherwise both decryption routines produce the same output. Suppose $s_i = 0$ and $s_i^\dagger = 1$, and suppose Dec does not output $\perp$. First, note that since $s_i \neq s_i^\dagger$, $\mathfrak{p}_i$ must be 1. Next, since Dec does not output $\perp$, it follows that $\mathsf{ct}_{i,0}$ is an encryption of $1\|w_i$, and w_i is a valid opening for certifying that com_i is a commitment to bit 0. Finally, let us see why $\mathsf{Dec}^\dagger$ set $s_i^\dagger = 1$: it decrypted $\mathsf{ct}_{i,1}$ and obtained $1\|w_i^\dagger$ such that $w_i^\dagger$ is a valid decommitment of com_i for bit 1. But this is a contradiction since we assumed perfect binding for the commitment scheme. The other case (where $s_i = 1$ and $s_i^\dagger = 0$) relies on the perfect correctness of PKE and is handled similarly.

2.2 IND-CCA PRC with low error rate.

In this section, we present our first CCA-PRC construction. This construction is adaptively α-robust, where $\alpha = \Theta(1/\lambda)$. In the next section, we discuss how to achieve constant rate robustness.

[5] Note that only the $\mathsf{sk}_{i,0}$ secret keys are included, the remaining are discarded.

[6] Here, we use the decryption-using-randomness property of the base PKE scheme.

Making KW ciphertexts pseudorandom: Note that the ciphertexts in the KW encryption scheme are neither pseudorandom, nor error tolerant. We first discuss how to make the ciphertexts pseudorandom. There are three obstacles:

1. *PKE ciphertexts.* The ciphertext components $c_{i,b}$ may not be pseudorandom. This can be fixed by requiring the underlying PKE scheme to have pseudorandom ciphertexts.
2. *Commitments.* The commitments used in the construction may also fail to be pseudorandom. This can be addressed by employing a commitment scheme with pseudorandom commitments. For instance, Naor's commitment scheme (based solely on the existence of a PRG) suffices.
3. *Verification key.* The most crucial issue is that the verification key vk, together with a valid signature, is never pseudorandom, since the verification algorithm itself acts as a distinguisher. However, Lamport's signature scheme provides signatures that are indistinguishable from random strings in the absence of vk. To hide vk, we can embed it inside one of the PKE encryptions, e.g., in both $c_{1,0}$ and $c_{1,1}$. Including it in both components is necessary for the security proof.

We note that CCA-secure schemes with pseudorandom ciphertexts have already been constructed in prior works.

Adding robustness: A first attempt is to replace the CPA-PKE ciphertexts in the previous construction with CPA-PRC ciphertexts. Since each of the ciphertexts are individually pseudorandom, and the payload is also pseudorandom, one can hope that this directly results in an $\mathsf{IND\text{-}CCA}$-secure PRC scheme. Unfortunately, this natural attempt is not compatible with the KW proof strategy. In particular, the decryptor must be able to detect if the PRC ciphertexts contain too much noise (otherwise the adversary can create ciphertexts which can detect the set of secret keys being used for decryption), and this is not possible if the decryptor recovers only half the randomness used for the PRC encoding.
We address this issue by using a two-layered approach — first encrypting the commitment openings using a PKE scheme, and then encrypting the resulting ciphertexts and commitments using a CPA-PRC scheme. The key generation algorithm samples $2n$ public/secret key pairs for the PKE scheme $\left(\mathsf{pke.pk}_{i,b}, \mathsf{pke.sk}_{i,b}\right)_{i,b}$. It also samples $2n$ PRC encoding/decoding keys $\left(\mathsf{prc.pk}_{i,b}, \mathsf{prc.sk}_{i,b}\right)$, and the public parameters $\mathsf{com.pp}$ for the commitment scheme,[7] and the public parameters $\mathsf{hprg.pp}$ for the hinting PRG scheme. We also sample a number ζ uniformly at random (from a sufficiently large field $\mathbb{F}$). The public and secret key for our CCA-PRC scheme are as follows:

$$\mathsf{PK} = \left(\zeta, \mathsf{com.pp}, \mathsf{hprg.pp}, \left(\mathsf{pke.pk}_{i,b}, \mathsf{prc.pk}_{i,b}\right)_{i,b}\right), \qquad \mathsf{SK} = \left(\mathsf{pke.sk}_{i,b}, \mathsf{prc.sk}_{i,b}\right)_{i,b}.$$

Encryption. The encryption algorithm samples a HPRG seed s and signing/verification keys $(\mathsf{ds.sk}, \mathsf{ds.vk})$. It computes $(r_0, r_1, \ldots, r_\lambda) = \mathsf{HPRG}(s)$ and parses $r_0 = (r_{00}, r_{01})$, where both r_{00}, r_{01} are λ bits long. Next, it samples commitment openings $(v_i)_i$, computes $\mathsf{com}_i = \mathsf{Com}(\mathsf{com.pp}, s_i, \mathsf{ds.vk}; v_i)$, and follows the KW template for computing the PKE ciphertexts. For each i, it sets

$$\begin{aligned} \mathsf{pke.ct}_{i,s_i} &= \mathsf{PKE.Enc}\left(\mathsf{pke.pk}_{i,s_i}, 1 \| v_i \| \mathsf{ds.vk}; r_i\right) \\ \mathsf{pke.ct}_{i,\overline{s_i}} &\leftarrow \mathsf{PKE.Enc}\left(\mathsf{pke.pk}_{i,\overline{s_i}}, 0^{2\lambda+1}\right) \end{aligned}$$

Next, it samples randomness $(z_{i,b})_{i,b}$ and computes PRC encodings of each ciphertext and commitment. That is, for each i, b, it sets

$$c_{i,b} = \mathsf{PRC.Encode}\left(\mathsf{prc.pk}_{i,b}, \left(\mathsf{pke.ct}_{i,b}, \mathsf{com}_i\right); z_{i,b}\right).$$

It then computes the payload c, which contains an encoding of the message m, randomness $(z_{i,b})_{i,b}$ used for PRC encoding, the commitments and the PKE ciphertexts. The payload is

$$c = \mathsf{PRG}(r_{00}) + \zeta \cdot \mathsf{ECC.Encode}\left(m, \left(z_{i,b}, \mathsf{pke.ct}_{i,b}, \mathsf{com}_i\right)_{i,b}\right).$$

[7] In our construction, we also require an additional property from the commitment scheme - the commitment to bit 0 should not depend upon the tag.

Here, PRG is a regular pseudorandom generator (with sufficient stretch) and the output of the PRG is a number in a large field $\mathbb{F}$, and ECC is a regular error correcting code (over the binary alphabet). The output of the ECC will be interpreted as a number in $\mathbb{F}$. Looking ahead, we will argue that with overwhelming probability over the choice of ζ, once the payload is fixed, the underlying components $\left(m, (z_{i,b}, \mathsf{pke.ct}_{i,b}, \mathsf{com}_i)_{i,b}\right)$ are fixed.

Finally, the encryption algorithm computes a signature on the payload and PRC codewords. It computes $\mathsf{ds}.\sigma = \mathsf{Sign}\left(\mathsf{ds.sk}, \left(c, (c_{i,b})_{i,b}\right)\right)$, and a masked version $\sigma = \mathsf{PRG}\left(r_{01}\right) \oplus \mathsf{ECC.Encode}(\mathsf{ds}.\sigma)$. The final ciphertext is $\left(c, (c_{i,b})_{i,b}, \sigma\right)$.

Information rate of our PRC scheme: Our PRC scheme preserves the information rate of the underlying error correcting code. Suppose ECC maps t bit messages to $k_{\mathrm{ecc}} \cdot t$ bits for some $k_{\mathrm{ecc}} > 1$, and let ℓ_{msg} denote the length of the message being encoded by our PRC scheme. The payload size is $k_{\mathrm{ecc}} \cdot (\ell_{\mathsf{msg}} + \mathsf{poly}(\lambda))$, since each CPA-PKE ciphertext encrypts $O(\lambda)$ bit strings, and the commitments are also $\mathsf{poly}(\lambda)$ bits long. The CPA-PRC codewords encode $\mathsf{poly}(\lambda)$ bit strings, and therefore have size $\mathsf{poly}(\lambda)$. Finally, the signature is only $O(\lambda)$ bits long. Putting everything together, the ciphertext size is dominated by $k_{\mathrm{ecc}} \cdot \ell_{\mathsf{msg}}$ (for long messages).

Decryption. The decryption algorithm uses the secret keys corresponding to the 0^n path (i.e., $(\mathsf{prc.sk}_{i,0}, \mathsf{pke.sk}_{i,0})_i$), and proceeds as follows:

- *Recovering a candidate seed.*
  1. First, it decrypts each $c_{i,0}$ using $\mathsf{prc.sk}_{i,0}$, recovering $\left(\underline{\mathsf{pke.ct}_{i,0}}, \underline{\mathsf{com}_i}\right)$. If any of the decryptions fail, it is set to a default value (but do not abort the decryption).
  2. Next, for each i, it decrypts $\mathsf{pke.ct}_{i,0}$. If the decrypted message is of the form $1 \| \underline{\mathsf{op}_i} \| \underline{\mathsf{ds.vk}_i}$ and $\underline{\mathsf{com}_i} = \mathsf{Com}\left(0, \underline{\mathsf{ds.vk}_i}; \underline{\mathsf{op}_i}\right)$, then it guesses that $t_i = 0$, else it guesses $t_i = 1$. Again, decryption does not abort if the decryption of $\mathsf{pke.ct}_{i,0}$ fails, it simply guesses $t_i = 1$.
- *Checking noise in the ciphertext components*
  1. Having obtained a candidate seed t, the decryption algorithm computes $(r_{00}, r_{01}, r_1, \ldots, r_\lambda) = \mathsf{HPRG}(t)$.
  2. It then recovers the (error-free) message, the randomness used for the PRC encodings, the commitments, and PKE ciphertexts from the payload. It computes $\left(\mathsf{msg}, (z_{i,b}, \mathsf{pke.ct}_{i,b}, \mathsf{com}_i)_i\right) = \mathsf{ECC.Decode}\left(\zeta^{-1}\left(\mathsf{PRG}\left(r_{00}\right) \oplus c\right)\right)$.
  3. Similarly, it recovers the error-free digital signature from σ — $\mathsf{ds}.\sigma = \mathsf{ECC.Decode}\left(\mathsf{PRG}\left(r_{01}\right) \oplus \sigma\right)$.
  4. It re-encrypts each PRC ciphertext using the error-free PKE ciphertexts and commitments, and the randomness obtained from the payload. For each i, b, it computes $c'_{i,b} = \mathsf{PRC.Encode}(\mathsf{prc.pk}_{i,b}, (\mathsf{pke.ct}_{i,b}, \mathsf{com}_i); z_{i,b})$ and checks that $\Delta\left(c_{i,b}, c'_{i,b}\right)$ is small (which ensures that the amount of noise in each PRC codeword is bounded).
  5. Similarly, it checks that the noise in c and σ is bounded. It then computes $c' = \mathsf{PRG}\left(r_{00}\right) \oplus \left(\mathsf{msg}, (z_{i,b}, \mathsf{pke.ct}_{i,b}, \mathsf{com}_i)_{i,b}\right)$, $\sigma' = \mathsf{PRG}\left(r_{01}\right) \oplus \mathsf{ds}.\sigma$, and checks that $\Delta\left(c, c'\right)$ and $\Delta\left(\sigma, \sigma'\right)$ are small.
- *Comparing the commitments and PKE ciphertexts recovered via decryption, with the commitments and PKE ciphertexts recovered from the payload.*
  1. For each i such that t_i is set to 0, we recovered a PKE ciphertext and commitment from $c_{i,0}$. The decryption algorithm checks that these are consistent with the PKE ciphertexts and commitments in the payload. If $t_i = 0$, we check that $\mathsf{pke.ct}_{i,0} = \underline{\mathsf{pke.ct}_{i,0}}$ and $\mathsf{com}_i = \underline{\mathsf{com}_i}$.
- *Performing the KW checks*
  1. For each i, it decrypts $\mathsf{pke.ct}_{i,t_i}$ using the randomness r_i, recovers $1 \| v_i \| \mathsf{ds.vk}_i$, checks that v_i is a valid opening of com_i for the bit s_i (w.r.t. $\mathsf{ds.vk}_i$), and that re-encryption of $1 \| v_i \| \mathsf{ds.vk}_i$ with randomness r_i gives $\mathsf{pke.ct}_{i,s_i}$. If any of these checks fail, it outputs $\perp$.
- *Checking the digital signature*: Finally, the decryption algorithm checks that all the verification keys $\mathsf{ds.vk}_i$ are equal, and that the signature $\mathsf{ds}.\sigma$ (obtained from σ) on $\left(c', \left(c'_{i,b}\right)_{i,b}\right)$ is valid (with respect to the common verification key).

Security of our construction. At a high level, our proof follows the KW proof structure. Let t^* denote the seed, and $\mathsf{ds.vk}^*$ the verification key used in the challenge ciphertext. First, using the security of the signature scheme, we can argue that for all decryption queries not close to the challenge ciphertext, if the verification key extracted from the ciphertext is equal to $\mathsf{ds.vk}^*$, then such decryption queries can be rejected. Next, we switch the commitment scheme's public parameters to make them equivocal at $\mathsf{ds.vk}^*$, this follows from the security of the commitment scheme. After this, we decrypt using the secret keys $\left(\mathsf{prc.sk}_{i,t_i^*}, \mathsf{pke.sk}_{i,t_i^*}\right)_i$. Using the perfect correctness of the PKE scheme, the perfect binding of the commitment scheme and adaptive robustness of the base PRC scheme, we argue that this switch in the decryption oracle goes unnoticed by the adversary. This indistinguishability proof is one of the key technical challenges in our proof, and we discuss this separately in the next paragraph. Assuming the decryption switch works, we then switch the PKE ciphertexts in the challenge ciphertext that are not on the t^* path. Instead of encrypting $\mathbf{0}$, we encrypt a commitment opening for bit $\overline{t_i^*}$ wrt $\mathsf{ds.vk}^*$ (recall, since our commitments are equivocal for tag $\mathsf{ds.vk}^*$, we can produce openings for both 0 and 1). This switch follows from the CPA security of the PKE scheme since the corresponding secret keys are not used in the decryption oracle. We now switch back the decryption oracle to use $(\mathsf{prc.sk}_{i,0}, \mathsf{pke.sk}_{i,0})_i$. At this point, note that the HPRG seed is only used to compute the randomness $(r_{00}, r_{01}, r_1, \ldots, r_\lambda)$. We can now use the HPRG security to switch all these to uniformly random, and then use the regular PRG security to make the payload c and signature component σ uniformly random. Next, we use the PRC security to replace the PRC ciphertexts $c_{i,1}$ with uniformly random strings (note that the corresponding PRC secret keys are not needed for decryption, and we already switched the payload c to a uniformly random string). At this point, the only components that are not uniformly random are the PRC ciphertexts $c_{i,0}$. We now switch the decryption oracle again, using the secret keys $(\mathsf{prc.sk}_{i,1}, \mathsf{pke.sk}_{i,1})_i$. Now, we can finally replace the codewords $(c_{i,0})_i$ with uniformly random strings. Therefore, all that remains is to show that if an adversary can distinguish between various modes of decryption, then this adversary can break adaptive robustness of the base PRC.

Switching the decryption oracle. Let Dec denote the regular decryption, and $\mathsf{Dec}^\dagger$ denote the decryption along path $\mathfrak{p}$. We want to argue that no p.p.t. adversary can produce a ciphertext $\mathsf{ct} = \left(c, (c_{i,b})_{i,b}, \sigma\right)$ such that the outputs of Dec and $\mathsf{Dec}^\dagger$ on ct are different (and this argument does not rely on the hiding of $\mathfrak{p}$). We will prove the following statements:

1. Suppose Dec finds a candidate seed t, and finally produces a non-$\perp$ message, then $\mathsf{Dec}^\dagger$ must also produce the same candidate t. [8]
2. If Dec and $\mathsf{Dec}^\dagger$ find the same candidate seed t, then they must produce the same output.

Let us consider the first statement. Suppose, on the contrary, Dec uses a candidate seed t and produces a non-$\perp$ output, and $\mathsf{Dec}^\dagger$ produces a different candidate seed $t^\dagger$ such that $t_i \neq t_i^\dagger$. As in the KW decryption switching proof, we will consider two sub-cases: $t_i = 0$ and $t_i^\dagger = 1$ and $t_i = 1$ and $t_i^\dagger = 0$.

- $t_i = 0$ and $t_i^\dagger = 1$: For the first sub-case, we will arrive at a contradiction by relying on the perfect binding of the commitments and the adaptive robustness. First, note that if $t_i \neq t_i^\dagger$, then $\mathfrak{p}_i$ must be equal to 1 (otherwise the strings t and $t^\dagger$ cannot differ at the i^{th} index. Next, from the structure of our payload, it follows that with all but negligible probability over the choice of ζ, there exists at most one r_{00}, $\mathsf{ds}.x = \left(\mathsf{msg}, (z_{i,b}, \mathsf{pke.ct}_{i,b}, \mathsf{com}_i)_i\right)$ such that $c = \mathsf{PRG}\left(r_{00}\right) + \zeta \cdot \mathsf{ECC}\left(\mathsf{ds}.x\right)$.
  Let $\left(\underline{\mathsf{pke.ct}_{i,0}}, \underline{\mathsf{com}_i}\right)$ be the ciphertext and commitment recovered from $c_{i,0}$. Since $t_i = 0$ and Dec does not output $\perp$, we get that
  . $\underline{\mathsf{com}_i} = \mathsf{com}_i$, $\underline{\mathsf{pke.ct}_{i,0}} = \mathsf{pke.ct}_{i,0}$
  . decryption of $\mathsf{pke.ct}_{i,0}$ produces $1 \| v_i \| \mathsf{ds.vk}_i$ such that v_i is an opening of com_i for bit 0
  . $\Delta\left(c_{i,0}, c'_{i,0}\right)$ and $\Delta\left(c_{i,1}, c'_{i,1}\right)$ are bounded

[8] We observe that the following is also true, i.e., suppose $\mathsf{Dec}^\dagger$ finds a candidate seed t, and finally produces a non-$\perp$ message, then Dec must also produce the same candidate t.

Next, we use the fact that $t_i^\dagger$ is set to 1. This means that $c_{i,1}$ decrypts to some $\left(\underline{\mathsf{pke.ct}_{i,1}}^\dagger, \underline{\mathsf{com}}_{i,1}^\dagger\right)$ and $\underline{\mathsf{com}}_i^\dagger$ is a commitment to 1 (with respect to some $\mathsf{ds.vk}^\dagger$). Now, if $\underline{\mathsf{com}}_i^\dagger = \mathsf{com}_i$, then we break the perfect binding property of the commitment scheme. This is because $\mathsf{com}_i = \underline{\mathsf{com}_i}$ is a commitment to the bit 0 (and commitments to the bit 0 do not depend on the tag), and also a commitment to the bit 1 wrt $\mathsf{ds.vk}^\dagger$ if $\underline{\mathsf{com}}_i^\dagger = \mathsf{com}_i$. On the other hand, if $\underline{\mathsf{com}}_i^\dagger \neq \mathsf{com}_i$, then we break the adaptive robustness of the base PRC scheme. This is because $c_{i,1}$ decrypts to $\left(\ldots, \underline{\mathsf{com}}_{i,1}^\dagger\right)$, and $c'_{i,1}$ (which is a PRC encryption of $(\mathsf{pke.ct}_{i,1}, \mathsf{com}_i)$ with randomness $z_{i,1}$) is close to $c_{i,1}$.

– $t_i = 1, t_i^\dagger = 0$: For the second sub-case, we will show a contradiction by relying on the perfect correctness of the PKE and the adaptive robustness of the PRC.

Next, let us consider the second case. If both Dec and $\mathsf{Dec}^\dagger$ recover the same candidate seed t and Dec outputs non-$\perp$, then $\mathsf{Dec}^\dagger$ must also output the same non-$\perp$ message. This again follows from the fact that c is binding to $(\mathsf{msg}, (z_{i,b}, \mathsf{pke.ct}_{i,b}, \mathsf{com}_i)_i)$, and since both recover the same guess t, both modes of decryption would output msg.

2.3 Improving the error rate of our CCA-PRC construction

One limitation of the previous construction is that it cannot handle errors at a constant fraction of the positions, as this would mean that some of the PRC ciphertexts can be completely lost. To achieve better error tolerance, the decryption algorithm must be able to recover the hinting PRG seed s even if some PRC components are corrupted. A natural fix is to use an error-correcting code: rather than feeding s directly into the hinting PRG, we encode it as $t = \mathsf{ECC.Encode}(s)$ and evaluate the HPRG on t. This way, even if several PRC ciphertexts are corrupted, the decryption algorithm can still reconstruct s and proceed with decryption. This gives us a PRC with constant rate adaptive robustness, assuming the base PRC and ECC have constant rate robustness.

Security proof. One nuance that arises is with respect to the security of hinting PRGs. Existing hinting PRG constructions prove security when the seed is sampled uniformly at random. However, it is easy to check that the proofs of security for standard hinting PRG constructions rely only on min-entropy of the input. Since ECC is a deterministic encoding, the entropy of t is identical to that of s. Hence, this modification is sound.

The security proof for the constant rate construction follows the same proof steps as the low-rate construction. The only step that requires some attention is the oracle switching step. To address this, we give a *lifting lemma* (in Lemma 12). This lemma essentially shows that if an adversary produces a ciphertext ct such that it can distinguish between Dec and $\mathsf{Dec}^\dagger$, then there exists an index i such that $\mathfrak{p}_i = 1$, $\tilde{t}_i$ (the noisy candidate seed recovered by Dec) is not equal to the corresponding bit $\widetilde{t_i^\dagger}$ recovered by $\mathsf{Dec}^\dagger$, and both $c_{i,0}$ and $c_{i,1}$ have bounded noise. Note that this is all that we required for the oracle switching proof for the low-rate scheme.

3 Preliminaries

Notations: Let PPT denote probabilistic polynomial-time. We denote the set of all positive integers up to n as $[n] := \{1, \ldots, n\}$. For any set S, we denote $\mathcal{P}(S)$ as the power set of S. We denote vectors/binary strings using bold font. For any two binary strings $\mathbf{x}, \mathbf{y}$, we use the notation $\mathbf{x}\|\mathbf{y}$ to denote $\mathbf{x}$ concatenated with $\mathbf{y}$. Here, x_i denotes the i^{th} bit of $\mathbf{x}$. For any $\mathbf{x} \in \mathbb{F}^n$, we denote $\mathsf{Ham}(\mathbf{x})$ as the Hamming weight of $\mathbf{x}$, i.e., the number of non-zero entries in $\mathbf{x}$. And, $\mathsf{Ham}(\mathbf{x}, \mathbf{y})$ denotes the Hamming distance between the binary strings $\mathbf{x}$ and $\mathbf{y}$, i.e., the number of indices where $\mathbf{x}$ and $\mathbf{y}$ differ. We also write $\Delta(\mathbf{x}, \mathbf{y})$ for Hamming distance. We denote $\mathcal{B}_\delta(\mathbf{x}, \mathbb{F}^n)$ as the set $\{\mathbf{x}' \in \mathbb{F}^n \mid \mathsf{Ham}(\mathbf{x}, \mathbf{x}') \leq \delta \cdot n\}$, i.e., the set of all strings with Hamming distance of r from $\mathbf{x}$. Furthermore, we define $\mathcal{B}_\delta(\mathbb{F}^n) := \mathcal{B}_\delta(\mathbf{0}, \mathbb{F}^n)$. We denote by $\mathsf{Ber}_\rho^{m \times n}$ the distribution over $m \times n$ matrices defined as follows: each entry is sampled independently, taking a value uniformly at random from $\mathbb{F} \setminus \{0\}$ with probability ρ, and 0 otherwise. For any finite set S, $x \leftarrow S$ denotes a uniformly random element

x from the set S. Similarly, for any distribution $\mathcal{D}$, $x \leftarrow \mathcal{D}$ denotes an element x drawn from the distribution $\mathcal{D}$.

Lemma 1 (Leftover Hash Lemma). *Let $\mathcal{D}$ be a distribution over $\mathbb{Z}_q^n$ with min-entropy k. For any $\epsilon > 0$ and $\ell \leq (k - 2\log(1/\epsilon) - O(1))/\log(q)$, the joint distribution of (C, Cs) where $C \leftarrow \mathbb{Z}_q^{\ell \times n}$ is uniformly random and $s \in \mathbb{Z}_q^n$ is drawn from the distribution $\mathcal{D}$ is ϵ close to the uniform distribution over $\mathbb{Z}_q^{\ell \times n} \times \mathbb{Z}_q^{\ell}$.*

3.1 Error-Correcting Codes

Definition 1 (Error-Correcting Code). *A binary error-correcting code is defined by three efficient algorithms* $(\mathsf{Setup}, \mathsf{Encode}, \mathsf{Decode})$ *with the following description.*

$\mathsf{Setup}(1^\gamma)$: *The setup algorithm takes as input the error-rate 1^γ and outputs a public parameter* pp.
$\mathsf{Encode}(\mathsf{pp}, \mathsf{msg})$: *The encoding algorithm takes as input a public parameter* pp *and a message* $\mathsf{msg} \in \{0,1\}^{\ell_{\mathsf{msg}}}$ *and outputs a codeword* $\mathsf{cw} \in \{0,1\}^{\ell_{\mathsf{cw}}}$.
$\mathsf{Decode}(\mathsf{pp}, \mathsf{cw})$: *The decoding algorithm takes as input a public parameter* pp *and a codeword* $\mathsf{cw} \in \{0,1\}^{\ell_{\mathsf{cw}}}$ *and outputs either a message* $\mathsf{msg} \in \{0,1\}^{\ell_{\mathsf{msg}}}$ *or* $\perp$.

An error-correcting code must satisfy the following properties.

- ***Correctness:*** *A code C is correct if for all* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\gamma)$, $\mathsf{msg} \in \{0,1\}^{\ell_{\mathsf{msg}}}$, $\mathsf{Decode}(\mathsf{pp}, \mathsf{Encode}(\mathsf{pp}, \mathsf{msg})) = \mathsf{msg}$.
- ***Robustness:*** *A code C is robust for up to γ fraction of errors if for all* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\gamma)$, $\mathsf{msg} \in \{0,1\}^{\ell_{\mathsf{msg}}}$ *and* $\mathbf{e} \in \{0,1\}^{\ell_{\mathsf{cw}}}$ *such that* $\mathsf{Ham}(\mathbf{e})$ *is less than* $\gamma \times \ell_{\mathsf{cw}}$*, it holds that* $\mathsf{Decode}(\mathsf{pp}, \mathsf{Encode}(\mathsf{pp}, \mathsf{msg}) + \mathbf{e}) = \mathsf{msg}$

The information rate of a code is the ratio between ℓ_{cw} and ℓ_{msg}.

Theorem 2 ([JST21]). *Fix $\gamma \in (0, 1/4)$. There exist binary error-correcting codes that are robust to error-rate γ with constant information rate.*

3.2 Pseudorandom Functions

Definition 2 (Pseudorandom Functions). *A family of pseudorandom functions is a pair of algorithms* $\mathsf{PRF} = (\mathsf{Setup}, \mathsf{Eval})$ *with the following description.*

- $\mathsf{Setup}(1^\lambda)$: *The key generation algorithm is a randomized algorithm that takes as input the security parameter 1^λ and outputs a key k.*
- $\mathsf{Eval}(k, x)$: *The evaluation algorithm is a deterministic algorithm that takes as input a key k and x and outputs y.*

Definition 3 (Security). *The PRF scheme* PRF *is secure if for all PPT adversary $\mathcal{A}$, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all $\lambda \in \mathbb{N}$,*

$$|\Pr[\mathcal{A}^{\mathsf{Eval}(k,\cdot)}(1^\lambda) = 1 \ : \ k \leftarrow \mathsf{Setup}(1^\lambda)] - \Pr[\mathcal{A}^{R(\cdot)}(1^\lambda) = 1 \ : \ R \leftarrow \mathcal{U}_\lambda]| \leq \mathsf{negl}(\lambda)$$

where $\mathcal{U}_\lambda$ is the set of all functions from the domain space of Eval *to the codomain space of* Eval.

3.3 Equivocal Statistically Binding Tag-based Set Commitments

We will follow the abstraction of set commitments as defined in [HKW20]. Here, the commitment algorithm takes as input a string $S \in \{0,1\}^N$ which represents the set $\{(i, S_i) : i \in [N]\}$.
Syntax. A tag-based set-commitment scheme for tag space $\mathcal{T}$, with commitment space $\mathcal{C}$ and opening space $\mathcal{O}$ is described by algorithms $\mathsf{Com} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open}, \mathsf{Equiv})$ with the following description.

$\mathsf{Setup}(1^\lambda, 1^N)$**:** The setup algorithm is a randomized algorithm that takes as input a unary representation of the security parameter λ and the set size N, and outputs public parameters pp.

$\mathsf{Commit}(\mathsf{pp}, S, \mathsf{tag})$**:** The commit algorithm is a randomized algorithm that takes as input a public parameter pp, a set-representation $S \in \{0,1\}^N$ and $\mathsf{tag} \in \mathcal{T}$, and outputs a commitment vector $(\mathsf{com}_1, \ldots, \mathsf{com}_N) \in \mathcal{C}^N$ and an opening vector $(\mathsf{op}_1, \ldots, \mathsf{op}_N) \in \mathcal{O}^N$.

$\mathsf{Open}(\mathsf{pp}, b, \mathsf{com}_i, \mathsf{tag}, \mathsf{op}_i)$**:** The opening algorithm is a deterministic algorithm that takes as input public parameters pp, a bit b, a commitment $\mathsf{com}_i \in \mathcal{C}$, $\mathsf{tag} \in \mathcal{T}$ and an opening $\mathsf{op}_i \in \mathcal{O}$ and outputs $\top/\bot$.

$\mathsf{Equiv}(1^\lambda, 1^N, \mathsf{tag}^*)$**:** The alternate setup algorithm takes as input the security parameter λ, the set size N and tag^*, and outputs public parameters pp together with commitment vector $(\mathsf{com}_i)_{i\in[N]}$ and openings $\left(\mathsf{op}_{i,0}, \mathsf{op}_{i,1}\right)_{i\in[N]}$.

An equivocal tag-based commitment scheme Com is *correct* if the following conditions hold:

- for all $\lambda, N \in \mathbb{N}$, $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^N)$, for all $\mathsf{tag} \in \mathcal{T}$ and $S \in \{0,1\}^N$, for all $\left((\mathsf{com}_i, \mathsf{op}_i)_{i\in[N]}\right) \leftarrow \mathsf{Commit}(\mathsf{pp}, S, \mathsf{tag})$, for all $i \in [N]$, $\mathsf{Open}(\mathsf{pp}, S_i, \mathsf{com}_i, \mathsf{tag}, \mathsf{op}_i) = \top$.
- for all $\lambda, N \in \mathbb{N}$, $\mathsf{tag}^* \in \mathcal{T}$, for all $\left(\mathsf{pp}, \left(\mathsf{com}_i, \mathsf{op}_{i,0}, \mathsf{op}_{i,1}\right)_i\right) \leftarrow \mathsf{Equiv}(1^\lambda, 1^N, \mathsf{tag}^*)$, for all $i \in [N]$ and $b \in \{0,1\}$, $\mathsf{Open}\left(\mathsf{pp}, b, \mathsf{com}_i, \mathsf{tag}^*, \mathsf{op}_{i,b}\right) = \top$.

Definition 4 (Statistical Binding). *A tag-based commitment scheme* Com *is statistical binding if for any adversary* $\mathcal{A}$ *and polynomial* p*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all* $\lambda \in \mathbb{N}$ *and* $N = p(\lambda)$*, the advantage of* $\mathcal{A}$ *in the following security game is bounded by* $\mathsf{negl}(\lambda)$*:*

1. $\mathcal{A}$ *chooses* tag^* *and sends it to the challenger.*
2. *The challenger computes* $\left(\mathsf{pp}, \left(\mathsf{com}_i, \mathsf{op}^*_{i,0}, \mathsf{op}^*_{i,1}\right)_i\right) \leftarrow \mathsf{Equiv}(1^\lambda, 1^N, \mathsf{tag}^*)$ *and sends* pp *to* $\mathcal{A}$*.*
3. $\mathcal{A}$ *outputs* $\mathsf{tag} \neq \mathsf{tag}^*, \mathsf{com}, \mathsf{op}_0, \mathsf{op}_1$ *and wins if* $\mathsf{Open}(\mathsf{pp}, b, \mathsf{com}, \mathsf{tag}, \mathsf{op}_b) = \top$ *for both* $b \in \{0,1\}$*.*

Definition 5 (Equivocality: Indistinguishability of Setup and Equiv). *A tag-based commitment scheme* Com *is said to be equivocal if for any PPT adversary* $\mathcal{A}$*, any* $\mathsf{tag}^* \in \mathcal{T}$ *and any* $S^* \in \{0,1\}^N$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all* $\lambda \in \mathbb{N}$ *and* n *that is polynomial in* λ*, the advantage of* $\mathcal{A}$ *is bounded by* $\mathsf{negl}(\lambda)$*.*

1. *The challenger samples* $b \leftarrow \{0,1\}$*.*
   If $b = 0$*, it generates* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$ *and* $\left((\mathsf{com}_i, \mathsf{op}_i)_{i\in[N]}\right) \leftarrow \mathsf{Commit}(\mathsf{pp}, S^*, \mathsf{tag}^*)$*. It sends* $(\mathsf{pp}, (\mathsf{com}_i, \mathsf{op}_i)_{i\in[N]})$ *to* $\mathcal{A}$*.*
   If $b = 1$ *it generates* $\left(\mathsf{pp}, \left(\mathsf{com}_i, \mathsf{op}_{i,0}, \mathsf{op}_{i,1}\right)_i\right) \leftarrow \mathsf{Equiv}(1^\lambda, 1^N, \mathsf{tag}^*)$*. It sends* $(\mathsf{pp}, \left(\mathsf{com}_i, \mathsf{op}_{i,S_i}\right)_{i\in[N]})$ *to* $\mathcal{A}$*.*
2. $\mathcal{A}$ *outputs* $b' \in \{0,1\}$ *and wins if* $b = b'$*, and the advantage of* $\mathcal{A}$ *is* $\Pr[\mathcal{A} \text{ wins}] - 1/2$*.*

Definition 6 (Pseudorandomness). *A tag-based commitment scheme* Com *is said to have pseudorandom commitments if, for any PPT adversary* $\mathcal{A}$ *and polynomial* p*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all* $\lambda \in \mathbb{N}$*,* $n = p(\lambda)$*,* $\mathcal{A}$ *wins the following game with at most* $\mathsf{negl}(\lambda)$ *advantage.*

1. *The challenger samples* $b \leftarrow \{0,1\}$*. It samples* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^N)$ *and sends* pp *to the adversary.*
2. *The adversary makes polynomially many queries. In each query, it sends a set* $S^{(i)}$ *and* $\mathsf{tag}^{(i)}$*.*
   - *If* $b = 0$*, challenger samples* $\left(\left(\mathsf{com}_j^{(i)}, \mathsf{op}_j^{(i)}\right)_{j\in[N]}\right) \leftarrow \mathsf{Commit}(\mathsf{pp}, S^{(i)}, \mathsf{tag}^{(i)})$ *and sends* $\left(\mathsf{com}_j^{(i)}\right)_{j\in[N]}$ *to* $\mathcal{A}$*.*
   - *If* $b = 1$*, challenger samples* $u_j^{(i)} \leftarrow \mathcal{C}$ *uniformly at random and sends* $\left(u_j^{(i)}\right)_{j\in[N]}$ *to* $\mathcal{A}$*.*
3. $\mathcal{A}$ *finally sends its guess* b' *and wins if* $b = b'$*. The advantage of* $\mathcal{A}$ *is* $\Pr[b = b'] - 1/2$*.*

Although tag-based commitment schemes were formally introduced in [HKW20], the concept was employed by Koppula and Waters [KW19] in their construction of chosen-ciphertext secure encryption schemes. In our work, we rely on the construction from [KW19] rather than the later construction in [HKW20]. The reason is that in [HKW20] the size of the commitment grows with the size of the set S, whereas in [KW19] the commitment size depends only on the fixed parameter N. This fixed-size property is crucial for our applications.

Theorem 3 ([Nao90], [KW19]). *Assuming the existence of secure PRGs, there exists equivocal, statistically binding, tag-based commitment schemes with pseudorandom commitments for tag space $\{0,1\}^\lambda$. The commitment space is $\{0,1\}^{4\cdot\ell}$ and the opening space is $\{0,1\}^\lambda$. Furthermore, for any pp sampled by Setup or Equiv, for any $S \in \{0,1\}^N$, for any $\mathsf{tag}, \mathsf{tag}'$, for any i such that $S_i = 0$, the following distributions are identical:*

$$\left\{\mathsf{com}_i \ : \ \left(\mathsf{com}_j, \mathsf{op}_j\right)_{j\in[N]} \leftarrow \mathsf{Commit}(\mathsf{pp}, S, \mathsf{tag})\right\} \equiv \left\{\mathsf{com}_i \ : \ \left(\mathsf{com}_j, \mathsf{op}_j\right)_{j\in[N]} \leftarrow \mathsf{Commit}(\mathsf{pp}, S, \mathsf{tag}')\right\}$$

3.4 Public-key Pseudorandom Codes

A public-key PRC with κ-correctness α-robustness is described by three algorithms $(\mathsf{Setup}, \mathsf{Encode}, \mathsf{Decode})$, satisfying the following criteria:

Syntax: There exist functions $s, p, \ell_{\mathsf{cw}}, \ell_{\mathsf{msg}}, r, \ell_{\mathsf{Enc}}, r'' : \mathbb{N} \to \mathbb{N}$ such that for all $\lambda \in \mathbb{N}$, $\mathsf{Setup} : 1^\lambda \times \{0,1\}^{r''(\lambda)} \to \{0,1\}^{p(\lambda)} \times \{0,1\}^{s(\lambda)}$, $\mathsf{Encode} : \{0,1\}^{p(\lambda)} \times \{0,1\}^{\ell_{\mathsf{msg}}(\lambda)} \times \{0,1\}^{\ell_{\mathsf{Enc}}(\lambda)} \to \{0,1\}^{\ell_{\mathsf{cw}}(\lambda)}$, and $\mathsf{Decode} : \{0,1\}^{s(\lambda)} \times \{0,1\}^{\ell_{\mathsf{cw}}(\lambda)} \times \{0,1\}^{r(\lambda)} \to \{0,1\}^{\ell_{\mathsf{msg}}(\lambda)} \cup \perp$.

Error correction, or robustness: For any $\lambda \in \mathbb{N}$, message $\mathsf{msg} \in \{0,1\}^{\ell_{\mathsf{msg}}(\lambda)}$,

$$\Pr\left[\mathsf{Decode}(\mathsf{sk}, \mathbf{c}+\mathbf{e}) \neq \mathsf{msg} \,\middle|\, \begin{array}{r}(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{Setup}(1^\lambda)\\ \mathbf{c} \leftarrow \mathsf{Encode}(\mathsf{pk}, \mathsf{msg})\\ \mathbf{e} \leftarrow \mathcal{B}_\alpha^{\ell_{\mathsf{cw}}}\end{array}\right] \ \leq \ \kappa(\lambda).$$

where the probability is over the randomness of $\mathsf{Setup}, \mathsf{Encode}$ and error e.

Pseudorandomness: For any polynomial-time adversary $\mathcal{A}$,

$$\Pr_{(\mathsf{pk},\mathsf{sk})\leftarrow\mathsf{Setup}(1^\lambda)}\left[\mathcal{A}^{\mathsf{O}_{\mathsf{pk}}(\cdot)}(1^\lambda, \mathsf{pk}) = 1\right] \ - \ \Pr_{\substack{(\mathsf{pk},\mathsf{sk})\leftarrow\mathsf{Setup}(1^\lambda)\\ U}}\left[\mathcal{A}^{U(1^\lambda,\cdot)}(1^\lambda, \mathsf{pk}) = 1\right] \ \leq \ \mathrm{negl}(\lambda),$$

where O_{pk} on any (even previously queried) input msg samples a random $r \in \{0,1\}^{\ell_{\mathsf{Enc}}(\lambda)}$ and outputs $\mathsf{Encode}(\mathsf{pk}, \mathsf{msg}; r)$ and U denotes an oracle that on any (even previously queried) input returns a fresh uniform string in $\{0,1\}^{\ell_{\mathsf{cw}}}$.

CCA-Security: For any stateful polynomial time adversary $\mathcal{A}$, we require the following to hold:

$$\Pr\left[b' = b \,\middle|\, \begin{array}{r}(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{Setup}(1^\lambda)\\ m \leftarrow \mathcal{A}^{\mathsf{Decode}(\mathsf{sk},\cdot)}(\mathsf{pk})\\ b \leftarrow \{0,1\}\\ \text{if } b = 0 : \mathsf{ct} \leftarrow \mathsf{Encode}(\mathsf{pk}, m)\\ \text{if } b = 1 : \mathsf{ct} \leftarrow \{0,1\}^{\ell_{\mathsf{cw}}(\lambda)}\\ b' \leftarrow \mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{ct})\end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

where

$$\mathcal{O}(\mathsf{ct}') = \begin{cases}\mathsf{Decode}(\mathsf{sk}, \mathsf{ct}') & \text{if } \forall \mathbf{e} : \mathsf{ct}' \oplus \mathbf{e} \neq \mathsf{ct}, \text{ where } \mathsf{Ham}(\mathbf{e}) \leq \alpha\ell_{\mathsf{cw}}(\lambda)\\ \perp & \text{otherwise .}\end{cases}$$

where $\ell_{\mathsf{cw}}(\lambda)$ is the length of the ciphertext and α is the robustness of the PRC.

The information rate of a PRC is the ratio between ℓ_{cw} and ℓ_{msg}.

Definition 7 (Soundness). *A public-key PRC with codeword space $\mathcal{Y}_\lambda$ is sound if* $\Pr\left[\mathsf{Decode}(\mathsf{sk},\mathbf{u}) \neq \perp \;\middle|\; (\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{Setup}(1^\lambda)\right.$ $\mathsf{negl}(\lambda)$. *For the binary syntax above, $\mathcal{Y}_\lambda = \{0,1\}^{\ell_{\mathsf{cw}}(\lambda)}$.*

Definition 8 (Adaptive Robustness). *A stronger notion of robustness called adaptive robustness is defined via the following game parameterized by a constant $\alpha \in (0, 1/2)$.*

1. *The challenger samples $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{Setup}(1^\lambda)$ and sends pk to the adversary.*
2. *The adversary sends (m, r, x) to the challenger.*
3. *The challenger computes $c = \mathsf{Encode}(\mathsf{pk}, m; r)$ and $m' \leftarrow \mathsf{Decode}(\mathsf{sk}, x)$. If $\mathsf{Ham}(c, x) \leq \alpha n$ and $m \neq m'$, then output the adversary wins. Otherwise, it loses.*

We say that a public-key pseudorandom code PRC is adaptively α-robust if, for any PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $\Pr[\mathcal{A}$ wins in the above game$] \leq \mathsf{negl}(\lambda)$.

Theorem 4 ([AAC+25]). *Assuming $2^{O(\sqrt{\lambda})}$ hardness of LPN, there exists an adaptively robust public key PRC with error rate $\delta/8$ for any $\delta \in (0, 1/4)$.*

If, in addition, the adversary is given the secret key sk in the first step of the above game, then the notion is referred to as *strong adaptive robustness* [CGG+25].

3.5 Hinting Pseudorandom Generator

A hinting PRG (HPRG) is a pair of algorithms $(\mathsf{Setup}, \mathsf{Eval})$ such that:

$\mathsf{Setup}(1^\lambda, 1^\ell)$: The generation algorithm is a randomized algorithm that takes as input a unary representation of the security parameter λ, a length parameter ℓ and outputs a public parameter pp and input length $n = n(\lambda, \ell)$.

$\mathsf{Eval}(\mathsf{pp}, s, i)$: The evaluation algorithm is a deterministic algorithm that takes as input a public parameter pp, a seed $s \in \{0,1\}^n$ and an index $i \in \{0\} \cup [n]$ and outputs a bit string $y \in \{0,1\}^\ell$.

Security: A hinting PRG scheme $(\mathsf{Setup}, \mathsf{Eval})$ is said to be secure if for any PPT adversary $\mathcal{A}$, polynomial $\ell(\cdot)$ there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, the following holds:

$$\left|\Pr\left[\beta \leftarrow \mathcal{A}\left(\mathsf{pp}, y_0^\beta, \left\{y_{i,b}^\beta\right\}\right) \;\middle|\; \begin{array}{r}(\mathsf{pp}, n) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell), s \leftarrow \{0,1\}^n, \\ \beta \leftarrow \{0,1\}, y_0^0 \leftarrow \{0,1\}^\ell, y_0^1 = \mathsf{Eval}(\mathsf{pp}, s, 0), \\ y_{i,b}^0 \leftarrow \{0,1\}^\ell \forall i \in [n], b \in \{0,1\}, \\ y_{i,s_i}^1 = \mathsf{Eval}(\mathsf{pp}, s, i), y_{i,\overline{s_i}}^1 \leftarrow \{0,1\}^\ell \forall i \in [n]\end{array}\right] - \frac{1}{2}\right| \leq \mathsf{negl}(\lambda)$$

Theorem 5 ([KW19, AP22]). *Assuming hardness of $\mathcal{X}$ where $\mathcal{X} \in \{\mathsf{LWE}, \mathsf{CDH}, \mathsf{DDH}, \mathsf{LHS}\}$, there exist secure hinting PRGs.*

We additionally require a security notion that we refer to as entropic security.

Entropic Security: A hinting PRG scheme $(\mathsf{Setup}, \mathsf{Eval})$ is said to be k-entropic secure if for any PPT adversary $\mathcal{A}$, polynomial $\ell(\cdot)$ there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, for all distributions $\mathcal{D}$ over $\{0,1\}^n$ such that $\mathcal{H}_\infty(\mathcal{D}) \geq k$ the following holds:

$$\left|\Pr\left[\beta \leftarrow \mathcal{A}\left(\mathsf{pp}, y_0^\beta, \left\{y_{i,b}^\beta\right\}\right) \;\middle|\; \begin{array}{r}(\mathsf{pp}, n) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell), s \leftarrow \mathcal{D}, \\ \beta \leftarrow \{0,1\}, y_0^0 \leftarrow \{0,1\}^\ell, y_0^1 = \mathsf{Eval}(\mathsf{pp}, s, 0), \\ y_{i,b}^0 \leftarrow \{0,1\}^\ell \forall i \in [n], b \in \{0,1\}, \\ y_{i,s_i}^1 = \mathsf{Eval}(\mathsf{pp}, s, i), y_{i,\overline{s_i}}^1 \leftarrow \{0,1\}^\ell \forall i \in [n]\end{array}\right] - \frac{1}{2}\right| \leq \mathsf{negl}(\lambda)$$

Remark 2 (On the instantiation of Entropic HPRGs). Both the CDH and LWE based constructions of hinting PRGs given by Koppula and Waters [KW19] remain secure even when the seed is not uniform but only has sufficiently high min-entropy. The only point in their analysis where the uniformity of the seed s is invoked is in a hybrid that applies a randomness-extraction style argument. In the CDH instantiation, this hybrid replaces $h = \prod_{k=1}^{n} g_{i,s_i}$ with a uniformly random group element h, and statistical closeness is justified using the Leftover Hash Lemma (see Lemma 1). Likewise, in the LWE construction, the analogous hybrid replaces $a = \sum_{k=1}^{n} a_{i,s_i}$ with a uniform vector a.
This argument remains valid verbatim when the seed s is sampled from any distribution s of sufficiently large min-entropy. Specifically, if $H_\infty(s) \geq m + 2\log(1/\varepsilon)$, where m is the number of bits that must be statistically close to uniform in the hybrid, then the Leftover Hash Lemma guarantees that the corresponding hybrid remains indistinguishable up to error ε. Consequently, by ensuring that the seed distribution provides this entropy margin, every hybrid in the Koppula-Waters proof continues to hold unchanged. Since both the CDH and LWE based instantiations in [KW19] implement the same extractor-like structure required for this step, their constructions immediately yield an *entropic* hinting PRG.
In the DDH-based construction of the hinting PRG by Alamati and Patranabis [AP22], the only place where the uniformity of the seed s is required is in establishing the pseudorandomness property of the generator. Somewhat surprisingly, their proof of the *hinting* security itself does not rely on any stronger guarantee about the distribution of s, such as full uniformity. In particular, they invoke an application of the Leftover Hash Lemma to argue that $([M], [Ms]) \approx_c ([M], [u])$, where u is uniformly distributed. Consequently, if the uniform seed is replaced by any distribution D with sufficiently high min-entropy so that the above extraction step continues to hold, the pseudorandomness property remains valid. With the parameters adjusted to satisfy this entropy requirement, their construction therefore yields an *entropic hinting PRG*.

Looking ahead, in our construction we will use an error-correction code applied on a randomly sampled k-bit string as our distribution $\mathcal{D}$. Given the correctness of the decoder of the error-correcting code, we will obtain as an output a string of length n which has at least min-entropy k. This fulfills the entropy requirement in the above definition.

3.6 Public Key Encryption

Definition 9 (Public-Key Encryption). *A public-key encryption scheme (PKE) is a triple of PPT algorithms* $\mathsf{PKE} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec})$*:*

$\mathsf{Setup}(1^\lambda, 1^n)$**:** *The setup algorithm takes as input a unary representation of the security parameter λ and the message length n and outputs a public key* pk *and secret key* sk*.*

$\mathsf{Enc}(\mathsf{pk}, m)$**:** *The encryption algorithm takes as input a public key* pk *and a message $m \in \{0,1\}^n$ and outputs a ciphertext* ct*.*

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$**:** *The decryption algorithm takes as input a secret key* sk *and a ciphertext* ct *and outputs either a message $m \in \{0,1\}^n$ or $\perp$.*

A scheme PKE *is* correct *if there exists a function $\epsilon(\lambda) = \mathsf{negl}(\lambda)$ such that for all $\lambda \in \mathbb{N}, n \in \mathbb{N}, m \in \{0,1\}^n$ and* pk, sk *in the support of* Setup*,*

$$\Pr[\mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, m)) = m] = 1 - \epsilon(\lambda),$$

where the probability is taken over the randomness of Enc*. We say that* PKE *is* perfectly *correct if $\epsilon(\lambda) = 0$. We say that a scheme is* almost perfectly correct *if there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all $\lambda, n \in \mathbb{N}$, for a $1 - \mathsf{negl}(\lambda)$ fraction of keys* pk, sk *in the support of* Setup*, for all $m \in \{0,1\}^n$, it holds that*

$$\Pr[\mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, m)) = m] = 1,$$

where the probability is taken over the randomness of Enc*.*

Theorem 6 ([DNR04b]). *Assuming the existence of PRGs, correct PKE implies almost perfectly correct PKE.*

Definition 10 (Decryption using Randomness). *A PKE scheme* $\mathsf{PKE} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec})$ *is* decryptable using encryption randomness *if there exists an algorithm* $\mathsf{Recover}$ *that takes as input the public key* pk*, a ciphertext* ct *and a randomness* r *and outputs a message* m*. This algorithm has the following property:*

$$\mathsf{Recover}(\mathsf{pk}, \mathsf{Enc}(\mathsf{pk}, m; r), r) = m$$

for all $\lambda \in \mathbb{N}, n \in \mathbb{N}, m \in \{0,1\}^n$ *and* pk, sk *in the support of* Setup*.*

Next, we define the IND-CPA security notion for public-key encryption.

Definition 11 ([GM84]). *A PKE scheme is said to be IND-CPA secure if for all stateful PPT adversaries* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all* $\lambda \in \mathbb{N}, n \in \mathbb{N}$*,*

$$\Pr\left[b' = b \;\middle|\; \begin{array}{r} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n) \\ (m_0, m_1) \leftarrow \mathcal{A}(\mathsf{pk}) \\ b \leftarrow \{0,1\} \\ \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b) \\ b' \leftarrow \mathcal{A}(\mathsf{ct}) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

3.7 Digital Signatures

A one-time signature scheme consists of the following algorithms:

$\mathsf{Setup}(1^\lambda)$**:** The setup algorithm takes as input a unary representation of the security parameter λ and outputs a verification key vk and a signing key sk.

$\mathsf{Sign}(\mathsf{sk}, m)$**:** The signing algorithm takes as input sk and a message $m \in \{0,1\}^*$ and outputs a signature $\sigma \in \{0,1\}^n$, where $n = n(\lambda)$ is the signature length.

$\mathsf{Verify}(\mathsf{vk}, m, \sigma)$**:** The verification algorithm takes as input vk, m, σ and outputs 1 (accept) or 0 (reject).

Correctness. The scheme is correct if for all $\lambda, \ell \in \mathbb{N}$, all $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{Setup}(1^\lambda)$, all $m \in \{0,1\}^\ell$, and $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$, we have

$$\mathsf{Verify}(\mathsf{vk}, m, \sigma) = 1,$$

except with negligible probability in λ over the coins of Setup and Sign. If the probability is 1, then the scheme is perfect correct.

Definition 12 (One-Time Strong Unforgeability). *Consider the following game between a challenger and an adversary* $\mathcal{A}$*:*

1. *The challenger samples* $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{Setup}(1^\lambda)$ *and gives* vk *to* $\mathcal{A}$*.*
2. $\mathcal{A}$ *makes one signing query* $m \in \{0,1\}^\ell$ *and receives* $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$*.*
3. $\mathcal{A}$ *outputs* $(m^\star, \sigma^\star)$*. It wins if* $\mathsf{Verify}(\mathsf{vk}, m^\star, \sigma^\star) = 1$ *and* $\sigma^\star \neq \sigma$*.*

The scheme is one-time strongly unforgeable if for all PPT $\mathcal{A}$*,* $\Pr[\mathcal{A}$ *wins the above game*$] \leq \mathsf{negl}(\lambda)$*.*

Theorem 7 ([Lam79]). *Assuming the existence of one-way functions and collision-resistant hash functions, there exists a one-time strong unforgeable digital signature scheme with* $|\sigma| = O(\lambda)$*.*

4 Candidate Robust PRC from Noisy-McEliece Type Assumptions

In this section, we propose a new family of hardness assumptions which give rise to adaptively robust pseudorandom codes. Recall the decisional McEliece assumption: let $\mathbf{G} \in \mathbb{F}_2^{m \times n}$ be a generator matrix of a binary Goppa code, let $\mathbf{\Pi} \leftarrow S_m$ [9] be a uniformly random permutation on m elements, and let $\mathbf{T} \leftarrow \mathsf{GL}_n(\mathbb{F}_2)$ be a uniformly random invertible matrix in $\mathbb{F}_2^{n \times n}$. The decisional McEliece assumption [McE78, DDMN12] then posits that the matrix

$$\mathbf{A} = \mathbf{\Pi} \cdot \mathbf{G} \cdot \mathbf{T}$$

is pseudorandom. Binary Goppa codes can only decode from a sublinear number of Hamming errors, specifically $O(n/\log(n))$ Hamming errors. Hence a tremendous research effort has been devoted to finding alternatives for binary Goppa codes in this construction. Notoriously, though, virtually all attempts to replace the binary Goppa code with other families of efficiently decodable linear codes have turned out to be insecure [Nie86, SS92, vT94, CC98, LS02, BLP08, FOPT10, GOT12, CGGU$^+$14, COTGU15, COT16, FOP$^+$16, CMCP17, KM23, CMT23, BC24, Mce26]. Furthermore, even the classical McEliece scheme was recently targeted by an algebraic distinguishing attack [Ran25]. A fundamental reason for the presence of such algebraic attacks is that the generator matrix $\mathbf{A}$ shares many algebraic invariants with the generator $\mathbf{G}$. We observe that all currently known algebraic attacks against McEliece-type assumptions fail if we only provide the matrix $\mathbf{A}$ masked by a suitable amount of Bernoulli noise. Indeed, the problem of solving algebraic systems of equations is mostly considered in the presence of exact equations. Hence, we propose the following general assumptions.

Assumption 1 (McEliece with Noise). *Let $\mathbb{F}$ be a finite field, let $m = m(\lambda)$, $k = k(\lambda)$, $n = n(\lambda)$ with $m \geq k \geq 2n > 0$ and $\ell = \ell(\lambda)$ be integers that are polynomial in λ, and let $\rho = \rho(\lambda) = O(1/\sqrt{n})$ and $\epsilon \in [0, 1]$ be a constant. Further let $(\mathsf{C}_\kappa)_{\kappa \in K}$, where $\mathsf{C}_\kappa \subseteq \mathbb{F}^m$, be a family of linear codes of length m and dimension k specified by a set $K = K(\lambda)$. For each $\kappa \in K$, let $\mathbf{G}_\kappa \in \mathbb{F}^{m \times k}$ be a generator matrix for C_κ. Let $\mathbf{\Pi} \leftarrow S_m$, $\kappa \leftarrow K$, let $\mathbf{T} \leftarrow (\mathbb{F}^{k \times 2n})_{full}$ be a uniformly random full-column-rank matrix, and let $\mathbf{E} \leftarrow \mathsf{Ber}_\rho^{m \times 2n}$. Let*

$$\mathbf{A} = \mathbf{\Pi} \cdot \mathbf{G}_\kappa \cdot \mathbf{T} + \mathbf{E}.$$

Now parse $\mathbf{A} = (\mathbf{A}_1 \| \mathbf{A}_2)$ where $\mathbf{A}_1, \mathbf{A}_2 \in \mathbb{F}^{m \times n}$. Further let $\mathbf{s}_1, \ldots, \mathbf{s}_\ell \leftarrow \mathcal{B}_\rho(\mathbb{F}^n)$ and $\mathbf{e}_1, \ldots, \mathbf{e}_\ell \leftarrow \mathcal{B}_\epsilon(\mathbb{F}^m)$. Then we conjecture that

$$(\mathbf{A}, \mathbf{A}_1\mathbf{s}_1 + \mathbf{e}_1, \ldots, \mathbf{A}_1\mathbf{s}_\ell + \mathbf{e}_\ell) \approx_c (\mathbf{A}, \mathbf{u}_1, \ldots, \mathbf{u}_\ell),$$

where the $\mathbf{u}_1, \ldots, \mathbf{u}_\ell \leftarrow \mathbb{F}^m$ are chosen uniformly at random. That is, we conjecture that the $\mathbf{A}_1\mathbf{s}_1 + \mathbf{e}_1, \ldots, \mathbf{A}_1\mathbf{s}_\ell + \mathbf{e}_\ell$ are pseudorandom given $\mathbf{A}$.

We remark that if a matrix $\mathbf{T}$ has t columns and $k \geq t + \lambda$, then a uniformly random matrix $\mathbf{T} \leftarrow \mathbb{F}^{k \times t}$ has full column rank except with negligible probability. In this case, we can drop the requirement that $\mathbf{T}$ be sampled conditioned on full column rank.

The choice of the code family $(\mathsf{C}_\kappa)_\kappa$ is part of the security assumption: the assumptions below should be read for explicitly specified, efficiently decodable families, not as consequences of generic coding-theoretic parameters. In particular, high dual distance is only a necessary sanity check against simple distinguishers, not a hardness guarantee. With this caveat, and for families that do not have an unusually small dual distance, we can, from a cryptanalytic perspective, make the following somewhat stronger assumption, which, similar to the decisional McEliece assumption, requires that the matrix $\mathbf{A} = \mathbf{\Pi} \cdot \mathbf{G}_\kappa \cdot \mathbf{T} + \mathbf{E}$ itself is pseudorandom.

Assumption 2 (Strong McEliece with Noise). *Let $\mathbb{F}$ be a finite field, let $m = m(\lambda)$, $k = k(\lambda)$, $n = n(\lambda)$ with $m \geq k \geq n > 0$ be integers that are polynomial in λ, and let $\rho = \rho(\lambda) = O(1/\sqrt{n})$. Further let $(\mathsf{C}_\kappa)_{\kappa \in K}$, where $\mathsf{C}_\kappa \subseteq \mathbb{F}^m$, be a family of linear codes of length m and dimension k specified by a set $K = K(\lambda)$. For each $\kappa \in K$, let $\mathbf{G}_\kappa \in \mathbb{F}^{m \times k}$ be a generator matrix for C_κ. Let $\mathbf{\Pi} \leftarrow S_m$, $\kappa \leftarrow K$, let $\mathbf{T} \leftarrow (\mathbb{F}^{k \times n})_{full}$ be a uniformly random full-column-rank matrix, and let $\mathbf{E} \leftarrow \mathsf{Ber}_\rho^{m \times n}$. Then we conjecture that the matrix*

$$\mathbf{A} = \mathbf{\Pi} \cdot \mathbf{G}_\kappa \cdot \mathbf{T} + \mathbf{E}$$

[9] We denote the symmetric group on $\{1, \ldots, m\}$ by S_m.

is pseudorandom in $\mathbb{F}^{m\times n}$*. That is, for any PPT distinguisher* $\mathcal{D}$ *it holds that* $|\Pr[\mathcal{D}(\mathbf{\Pi}\cdot\mathbf{G}_\kappa\cdot\mathbf{T}+\mathbf{E})=1]-\Pr[\mathcal{D}(\mathbf{U})=1]|<\mathsf{negl}(\lambda)$*, where* $\mathbf{U}\leftarrow\mathbb{F}^{m\times n}$ *is uniformly random.*

We do not present the strong McEliece with Noise assumption as a new route to ordinary public-key encryption. In our use of this assumption, it is combined with the standard $1/\sqrt{n}$-noise LPN assumption to obtain the McEliece with Noise assumption above, and LPN is already known to imply public-key encryption [Ale03].

Candidate code families and limitations. We view the noisy McEliece assumptions as candidate assumption families rather than as fully cryptanalyzed concrete proposals. A plausible instantiation should use a code family with efficient decoding, good relative distance, sufficiently large dual distance, and no known algebraic structure that remains exploitable after coordinate permutation and the addition of noise. Our main candidate is obtained by taking suitable algebraic-geometric codes over $\mathbb{F}_{16}$, for example from the Garcia–Stichtenoth tower, and expanding them to binary codes. Over $\mathbb{F}_{16}$, this tower has genus-to-length ratio tending to $1/3$; by the standard AG-code bounds, choosing a divisor of relative degree $3/4$ yields asymptotically good families with rate $R\geq 3/4-1/3=5/12$, relative distance $\delta\geq 1/4$, dual-distance bound $\delta^\perp\geq 3/4-2/3=1/12$, and efficient unique decoding from half the designed distance, i.e., a $1/8$ fraction of errors [GS95, Sti09]. After binary expansion these constants degrade by at most a constant factor.
Concretely, for an $\mathbb{F}_{16}$-linear code C one can write each symbol as a vector in $\mathbb{F}_2^4$, optionally composing the embedding in each coordinate with a public random invertible $\mathbb{F}_2$-linear map. The McEliece permutation is then applied bit-wise rather than symbol-wise. This is intended to obscure the extension-field structure used by Schur-product and square-code attacks against algebraic-code variants of McEliece [CGGU$^+$14, CMCP17].
A second natural cryptanalytic target is given by the binary Raw Reed–Solomon codes of [SKS19]. These codes have very strong distance properties and polynomially large dual distance, and they share the same high-level rationale of viewing an algebraic code over an extension field as a binary code. However, because they remain Reed–Solomon-derived, they are close to regimes where square-code and Schur-product distinguishers are especially relevant. We therefore regard Raw Reed–Solomon codes as an important target for further cryptanalysis rather than as our primary proposed instantiation. More generally, further work is needed to analyze whether noisy permuted binary expansions resist adaptations of known algebraic attacks, including the attacks and parameter regimes discussed in [CGG$^+$25].
One basic necessary condition is that the dual code of C_κ does not contain non-zero vectors of very low Hamming weight. In particular, if there are vectors of constant Hamming weight in the dual, then a simple polynomial-time distinguisher exists. To avoid this basic vulnerability, we require the codes used in any proposed instantiation to have high dual distance, following [CGG$^+$25], who impose a similar condition to guarantee the absence of low-weight parity constraints and to rule out low-degree statistical or algebraic distinguishers in their assumptions. For completeness, we briefly describe the attack that arises when this condition is not enforced.
Assume that the dual distance $\bar{d}$ of the code generated by $\mathbf{G}_\kappa$ is very small (e.g. constant). Let $\mathbf{H}$ be a parity check matrix for this code, i.e., it holds that $\mathbf{H}\cdot\mathbf{G}_\kappa=0$. Then there is a vector $\mathbf{t}$ in the row-span of $\mathbf{H}$ with Hamming weight $\bar{d}$. The attacker does not know $\mathbf{H}$ or $\mathbf{t}$, but can enumerate all public-coordinate vectors $\mathbf{h}\in\mathbb{F}^m$ of Hamming weight $c=\bar{d}$. Since $\mathbf{\Pi}$ is a permutation, this enumeration includes $\mathbf{h}=\mathbf{t}\cdot\mathbf{\Pi}^{-1}$ and takes time m^c for constant c. For this vector, the attacker calculates:

$$\begin{aligned}\mathbf{h}\cdot\mathbf{A}&=\mathbf{h}\cdot(\mathbf{\Pi}\cdot\mathbf{G}_\kappa\cdot\mathbf{T}+\mathbf{E})\\&=\mathbf{h}\cdot\mathbf{\Pi}\cdot(\mathbf{G}_\kappa\cdot\mathbf{T}+\mathbf{\Pi}^{-1}\cdot\mathbf{E})\\&=\mathbf{t}\cdot(\mathbf{G}_\kappa\cdot\mathbf{T}+\mathbf{\Pi}^{-1}\cdot\mathbf{E})\\&=\underbrace{\mathbf{t}\cdot\mathbf{G}_\kappa\cdot\mathbf{T}}_{=0}+\underbrace{\mathbf{t}\cdot\mathbf{\Pi}^{-1}\cdot\mathbf{E}}_{=:\mathbf{h}'}\end{aligned}$$

Indeed, if $\mathsf{Ham}(\mathbf{t})=c$, then each coordinate of $\mathbf{t}\cdot\mathbf{\Pi}^{-1}\mathbf{E}$ is nonzero with probability at most $c\rho$, and hence the resulting vector has sublinear weight with overwhelming probability when c is constant and $\rho=O(1/\sqrt{n})$. By contrast, for a random matrix and any non-zero choice of $\mathbf{h}$, the output of this procedure has high weight with overwhelming probability. Therefore, the attacker easily distinguishes between the two cases.

We can also show that the strong McEliece with Noise assumption implies the McEliece with Noise assumption under the additional standard $1/\sqrt{n}$-noise LPN assumption.

Lemma 2. *Assume that the strong McEliece with Noise assumption and the $1/\sqrt{n}$-noise LPN assumption hold. Then the McEliece with Noise assumption holds.*

Proof (Sketch). The proof is a simple hybrid argument. First, we use the strong McEliece with Noise assumption to replace the matrix $\mathbf{A}$ with a uniformly random matrix. Then, in the following hybrids, we successively replace the values $\mathbf{A}_1\mathbf{s}_i + \mathbf{e}_i$ by uniformly random $\mathbf{u}_i$, with $\mathbf{A} \coloneqq (\mathbf{A}_1 \| \mathbf{A}_2)$.
For the latter step we use a standard consequence of low-noise LPN. Starting from LPN with a uniform secret and Bernoulli noise of rate $\rho = O(1/\sqrt{n})$, one first puts the sample matrix in systematic form and applies the corresponding parity-check transformation. This eliminates the original secret and yields an LPN instance in which both the new secret and the new error are distributed according to Ber_ρ. Since our assumption uses a constant error rate ϵ and $\rho = o(1)$, we can add an independent smoothing error to the right-hand side, chosen so that the sum of the low-noise error and the smoothing error is distributed as Ber_ϵ; this operation is oblivious and preserves indistinguishability. Finally, the leakage lemma of [Döt15, Lemma 6] allows us to reveal the Hamming weights of the new secret and of the smoothed error, losing only a polynomial factor. Conditioning on these weights gives the fixed-weight, and hence Hamming-ball, distributions used here. The cited lemma is stated over $\mathbb{F}_2$, but the argument is linear-algebraic and extends directly to any fixed finite field. □

Construction We will now show that the noisy McEliece assumption for a suitable family of codes $(\mathsf{C}_\kappa)_\kappa$, together with the standard $1/\sqrt{n}$-noise LPN assumption, gives rise to adaptively robust pseudorandom codes. Importantly, this PRC construction enjoys essentially the same robustness properties as C_κ.
Now fix a family of codes $(\mathsf{C}_\kappa)_\kappa$ which can efficiently decode from an α fraction of errors. Fix constants $\epsilon, \delta > 0$. Let $m = O(n)$ and ρ be such that $4\rho^2 \cdot n + \epsilon + \delta \leq \alpha$ (e.g. $\rho = \sqrt{(\alpha - \epsilon - \delta)/(4n)} = O(1/\sqrt{n})$). For soundness, we additionally require $2n + H_q(\alpha)m \leq m - \omega(\log_q \lambda)$, where $q = |\mathbb{F}|$ and H_q denotes the q-ary entropy function. The pseudorandom code $\mathsf{PRC} = (\mathsf{Setup}, \mathsf{Encode}, \mathsf{Decode})$ is given as follows.

- $\mathsf{Setup}(1^\lambda)$: Choose $\mathbf{\Pi} \leftarrow S_m$, $\kappa \leftarrow K$ and $\mathbf{T} \leftarrow (\mathbb{F}^{k \times 2n})_{\text{full}}$ uniformly at random, and sample $\mathbf{E} \leftarrow \mathsf{Ber}_\rho^{m \times 2n}$. Set $\mathbf{A} = \mathbf{\Pi}\mathbf{G}_\kappa\mathbf{T} + \mathbf{E}$. Output $\mathsf{pk} = \mathbf{A}$ and $\mathsf{sk} = (\mathbf{\Pi}, \kappa, \mathbf{T})$.
- $\mathsf{Encode}(\mathsf{pk} = \mathbf{A}, \mathsf{msg} \in \mathcal{B}_\rho(\mathbb{F}^n))$: Sample $\mathbf{s} \leftarrow \mathcal{B}_\rho(\mathbb{F}^n)$ uniformly at random, set $\mathbf{x} = (\mathbf{s}, \mathsf{msg}) \in \mathcal{B}_\rho(\mathbb{F}^{2n})$ and sample $\mathbf{e} \leftarrow \mathcal{B}_\epsilon(\mathbb{F}^m)$. Compute and output $\mathbf{c} = \mathbf{A} \cdot \mathbf{x} + \mathbf{e}$.
- $\mathsf{Decode}(\mathsf{sk} = (\mathbf{\Pi}, \kappa, \mathbf{T}), \tilde{\mathbf{c}})$: Compute $\mathbf{c}' = \mathbf{\Pi}^{-1}\tilde{\mathbf{c}}$ and then $\mathbf{y} = \mathsf{C}_\kappa.\mathsf{Decode}(\mathbf{c}')$. If this outputs $\perp$, output $\perp$. If $\mathsf{Ham}(\mathbf{c}' - \mathbf{G}_\kappa\mathbf{y}) > \alpha m$, output $\perp$. Otherwise, compute the unique $\mathbf{x}$ such that $\mathbf{T} \cdot \mathbf{x} = \mathbf{y}$, parse $\mathbf{x} = (\mathbf{s}, \mathsf{msg})$ and check if $\mathbf{s} \in \mathcal{B}_\rho(\mathbb{F}^n)$ and $\mathsf{msg} \in \mathcal{B}_\rho(\mathbb{F}^n)$. If any of these steps fail, output $\perp$; otherwise, output msg.

We note that in the above construction, we require the message msg to be sparse. To encode a dense message $\mathsf{msg}' \in \mathbb{F}^\ell$, given ρ, we choose n such that $|\mathbb{F}^\ell| \leq |\mathcal{B}_\rho(\mathbb{F}^n)|$.

Pseudorandomness The pseudorandomness property of PRC follows immediately from (in fact, is equivalent to) the noisy McEliece assumption.

Lemma 3. *The code PRC has pseudorandom codewords, given that the noisy McEliece assumption holds.*

Soundness We first record that the construction satisfies the usual soundness property: a uniformly random word is rejected except with negligible probability.

Lemma 4. *Assume that $2n + H_q(\alpha)m \leq m - \omega(\log_q \lambda)$, where $q = |\mathbb{F}|$ and H_q is the q-ary entropy function. Then PRC is sound.*

Proof. Fix any secret key $\mathsf{sk} = (\mathbf{\Pi}, \kappa, \mathbf{T})$. Since $\mathbf{G}_\kappa$ has rank k and $\mathbf{T}$ has full column rank, the effective code

$$\mathsf{C}_{\kappa,\mathbf{T}} = \{\mathbf{\Pi}\mathbf{G}_\kappa\mathbf{T}\mathbf{x} : \mathbf{x} \in \mathbb{F}^{2n}\}$$

has size q^{2n}. By the explicit distance check in Decode, if a word $\mathbf{u} \in \mathbb{F}^m$ is accepted, then $\mathbf{u}$ lies in an αm-Hamming ball around some codeword in $\mathsf{C}_{\kappa,\mathbf{T}}$. Hence

$$\Pr_{\mathbf{u}\leftarrow\mathbb{F}^m}[\mathsf{Decode}(\mathsf{sk},\mathbf{u}) \neq \perp] \leq \frac{|\mathsf{C}_{\kappa,\mathbf{T}}| \cdot |\mathcal{B}_\alpha(\mathbb{F}^m)|}{q^m} \leq q^{2n+H_q(\alpha)m-m} \leq q^{-\omega(\log_q \lambda)} = \mathsf{negl}(\lambda).$$

The bound is uniform over the choice of sk, and therefore also holds over the randomness of Setup. □

Adaptive Robustness We will now argue adaptive robustness of PRC. We will use the following simple facts about Bernoulli distributed matrices in the proof. For a matrix $\mathbf{M} = (\mathbf{m}_1, \ldots, \mathbf{m}_N) \in \mathbb{F}^{m\times N}$, we define the induced matrix weight $\mathsf{Ham}(\mathbf{M})$ via $\mathsf{Ham}(\mathbf{M}) = \max_{i\in[N]} \mathsf{Ham}(\mathbf{m}_i)$. This matrix weight is sub-multiplicative, that is, it holds for every $\mathbf{x} \in \mathbb{F}^N$ that $\mathsf{Ham}(\mathbf{M}\cdot\mathbf{x}) \leq \mathsf{Ham}(\mathbf{M}) \cdot \mathsf{Ham}(\mathbf{x})$. If $\mathbf{E} \leftarrow \mathsf{Ber}_\rho^{m\times N}$, then a simple Chernoff bound and a union bound imply that $\mathsf{Ham}(\mathbf{E}) \leq 2\rho m$ except with probability $N \cdot e^{-\frac{1}{3}\rho m}$, which is negligible for our parameter choices.

Lemma 5. *Assume that C_κ can efficiently uniquely decode from up to an $\alpha < 1/4$ fraction of Hamming errors and that $4\rho^2 n + \epsilon + \delta \leq \alpha$. Then PRC is a strong adaptively robust PRC against a δ-fraction of adversarial errors, except with negligible probability over the choice of the public key $\mathbf{A}$.*

Proof. As discussed above, it follows by a Chernoff bound and a union bound that $\mathsf{Ham}(\mathbf{E}) \leq 2\rho m$, except with negligible probability $2n \cdot e^{-\frac{1}{3}\rho m}$. Hence, fix a public key $\mathsf{pk} = \mathbf{A}$ where $\mathbf{A} = \mathbf{\Pi} \cdot \mathbf{G}_\kappa \cdot \mathbf{T} + \mathbf{E}$, such that $\mathsf{Ham}(\mathbf{E}) \leq 2\rho m$. Consider a codeword

$$\begin{aligned}\mathbf{c} &= \mathbf{A}\mathbf{x} + \mathbf{e}\\ &= \mathbf{\Pi}\mathbf{G}_\kappa\mathbf{T}\mathbf{x} + \mathbf{E}\mathbf{x} + \mathbf{e}.\end{aligned}$$

It holds that

$$\mathsf{Ham}(\mathbf{E}\mathbf{x}+\mathbf{e}) \leq \mathsf{Ham}(\mathbf{E}) \cdot \mathsf{Ham}(\mathbf{x}) + \mathsf{Ham}(\mathbf{e}) \leq 2\rho \cdot m \cdot 2\rho \cdot n + \epsilon \cdot m = (4\rho^2 n + \epsilon)m.$$

As we require $4\rho^2 n + \epsilon \leq \alpha - \delta$, it follows that PRC is adaptively robust against errors of weight at most $\delta \cdot m$: fix any adversarial error $\mathbf{e}^*$ with Hamming weight at most $\delta \cdot m$. Then

$$\hat{\mathbf{c}} = \mathbf{c} + \mathbf{e}^* = \mathbf{\Pi}\mathbf{G}_\kappa\mathbf{T}\mathbf{x} + \mathbf{E}\mathbf{x} + \mathbf{e} + \mathbf{e}^* = \mathbf{\Pi}\mathbf{G}_\kappa\mathbf{T}\mathbf{x} + \mathbf{d},$$

where we set $\mathbf{d} = \mathbf{E}\mathbf{x} + \mathbf{e} + \mathbf{e}^*$. As $\mathsf{Ham}(\mathbf{E}\mathbf{x}+\mathbf{e}) \leq (\alpha-\delta)m$ and $\mathsf{Ham}(\mathbf{e}^*) \leq \delta m$, it follows by the triangle inequality that $\mathsf{Ham}(\mathbf{d}) \leq \alpha m$. Hence

$$\begin{aligned}\mathbf{c}' &= \mathbf{\Pi}^{-1}(\hat{\mathbf{c}})\\ &= \mathbf{G}_\kappa\mathbf{T}\mathbf{x} + \mathbf{\Pi}^{-1}\mathbf{d}.\end{aligned}$$

Thus we get that $\mathbf{c}'$ has distance at most αm from C_κ as $\mathsf{Ham}(\mathbf{\Pi}^{-1}\mathbf{d}) = \mathsf{Ham}(\mathbf{d}) \leq \alpha m$. Hence $\mathsf{C}_\kappa.\mathsf{Decode}$ will uniquely recover $\mathbf{T}\mathbf{x}$. Since $\mathbf{T}$ has full column rank, the final linear solve recovers $\mathbf{x}$ and therefore msg. □

5 CCA secure Pseudorandom Codes

In this section, we present our CCA PRC transformation. In general, constructing PRCs becomes easier when the underlying field is large [DMR25]. As in prior work, we focus exclusively on the binary case for clarity of presentation. Nevertheless, we expect that our transformation extends naturally to arbitrary finite fields, provided that the underlying primitives are replaced by their corresponding counterparts over such fields, for example by relying on the noisy McEliece assumption (Section 4) defined over general finite fields.

Our construction requires the following primitives:

- $\mathsf{DS} = (\mathsf{DS.Setup}, \mathsf{DS.Sign}, \mathsf{DS.Verify})$: a one-time strong unforgeable signature scheme with message space $\{0,1\}^*$, verification key space $\{0,1\}^{\ell_{\mathsf{vk}}}$ and signature space $\{0,1\}^{\lambda}$
- $\mathsf{Com} = (\mathsf{Com.Setup}, \mathsf{Com.Commit}, \mathsf{Com.Open}, \mathsf{Com.Equiv})$: a statistically binding equivocal tag-based set-commitment scheme with pseudorandom commitments, with tag space $\{0,1\}^{\ell_{\mathsf{vk}}}$, commitment space $\{0,1\}^{\ell_{\mathrm{com}}}$ and opening space $\{0,1\}^{\ell_{\mathrm{op}}}$.
- $\mathsf{PKE} = (\mathsf{PKE.Setup}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$: an almost perfect-correctness IND-CPA secure PKE with message space $\{0,1\}^{\ell_{\mathrm{op}}+\ell_{\mathsf{vk}}+1}$ and randomness $\{0,1\}^{\lambda}$. The ciphertext size is $\ell_{\mathrm{pke.ct}}$.
- $\mathsf{ECC} = (\mathsf{ECC.Setup}, \mathsf{ECC.Encode}, \mathsf{ECC.Decode})$: a γ-robust error-correcting code over alphabet $\{0,1\}$, with message-rate c_{ecc} (that is, for binary strings of length k, it produces codewords of length $c_{\mathrm{ecc}} \cdot k$)
- $\mathsf{HPRG} = (\mathsf{HPRG.Setup}, \mathsf{HPRG.Eval})$: a hinting pseudorandom generator that takes $n = c_{\mathrm{ecc}} \cdot \lambda$ bits as input, and outputs $n+2$ blocks where each block of the output has λ bits
- $\mathsf{PRF} = (\mathsf{PRF.Setup}, \mathsf{PRF.Eval})$: a pseudorandom function with key space, input and output space $\{0,1\}^{\lambda}$.
- PRG: a (regular) pseudorandom generator that maps λ bits to ℓ_{PRG} bits, where $\ell_{\mathrm{PRG}} = 2c_{\mathrm{ecc}} \cdot (\ell_{\mathsf{msg}} + \lambda + 2n \cdot \ell_{\mathrm{pke.ct}} + n \cdot (\ell_{\mathrm{com}})) + 3\lambda$. We will interpret the output as a number in $\mathbb{F}_{2^{\ell_{\mathrm{PRG}}}}$.
- $\mathsf{PRC} = (\mathsf{PRC.Setup}, \mathsf{PRC.Encode}, \mathsf{PRC.Decode})$: an α adaptively-robust CPA secure public-key pseudorandom code with message space $\{0,1\}^{\ell_{\mathrm{pke.ct}}+\ell_{\mathrm{com}}}$, randomness $\{0,1\}^{\lambda}$. The codeword length is $\ell_{\mathsf{cw}} = \ell_{\mathrm{PRG}}/(2 \cdot n)$.[10]

$\mathsf{Setup}(1^{\lambda})$**:** The setup algorithm performs the following steps:

1. Sample $\mathsf{ecc.pp} \leftarrow \mathsf{ECC.Setup}(1^{\gamma})$.
2. Sample $\mathsf{hprg.pp} \leftarrow \mathsf{HPRG.Setup}(1^{\lambda})$.
3. Sample $2n$ different PKE keys, i.e., $(\mathsf{pke.pk}_{i,b}, \mathsf{pke.sk}_{i,b}) \leftarrow \mathsf{PKE.Setup}(1^{\lambda})$ for all $b \in \{0,1\}, i \in [n]$.
4. Sample $2n$ different PRC keys, i.e., $(\mathsf{prc.pk}_{i,b}, \mathsf{prc.sk}_{i,b}) \leftarrow \mathsf{PRC.Setup}(1^{\lambda})$ for all $b \in \{0,1\}, i \in [n]$.
5. Sample $\mathsf{com.pp} \leftarrow \mathsf{Com.Setup}\left(1^{\lambda}, 1^{n}\right)$.
6. Sample $B_{\mathsf{pl}} \leftarrow \{0,1\}^{\ell_{\mathrm{PRG}}}$.
7. Output the public key $\mathsf{pk} = (\mathsf{ecc.pp}, \mathsf{hprg.pp}, B_{\mathsf{pl}}, \mathsf{com.pp}, (\mathsf{pke.pk}_{i,b}, \mathsf{prc.pk}_{i,b})_{i\in[n],b\in\{0,1\}})$ and the secret key $\mathsf{sk} = (\mathsf{pke.sk}_{i,0}, \mathsf{prc.sk}_{i,0})_{i\in[n]}$.

$\mathsf{Enc}(\mathsf{pk}, m)$**:** Parse the public key $\mathsf{pk} = (\mathsf{ecc.pp}, \mathsf{hprg.pp}, B_{\mathsf{pl}}, \mathsf{com.pp}, (\mathsf{pke.pk}_{i,b}, \mathsf{prc.pk}_{i,b})_{i\in[n],b\in\{0,1\}})$.

1. Sampling the various random strings needed for encryption:
   (a) Sample $s \leftarrow \{0,1\}^{\lambda}$, set $t = \mathsf{ECC.Encode}(\mathsf{ecc.pp}, s)$.
   (b) Sample $(\mathsf{ds.vk}, \mathsf{ds.sk}) \leftarrow \mathsf{DS.Setup}(1^{\lambda})$.
   (c) Sample $\mathsf{K} \leftarrow \mathsf{PRF.Setup}(1^{\lambda})$.
   (d) For each $i \in [n]$, choose $\tilde{r}_i \leftarrow \{0,1\}^{\lambda}$.
   (e) Let $(r_{00}, r_{01}, r_1, \ldots, r_n) = \mathsf{HPRG.Eval}(\mathsf{hprg.pp}, t)$.
2. Compute the PKE ciphertexts and the commitments.
   First compute $(\mathsf{com}_i, \mathsf{op}_i)_i \leftarrow \mathsf{Commit}(\mathsf{com.pp}, t, \mathsf{ds.vk})$ (here, $\mathsf{ds.vk}$ is the tag used for the commitment).
   For each $i \in [n]$,
   $$\mathsf{pke.ct}_{i,t_i} := \mathsf{PKE.Enc}\left(\mathsf{pke.pk}_{i,t_i}, 1 \| \mathsf{op}_i \| \mathsf{ds.vk}; r_i\right)$$
   $$\mathsf{pke.ct}_{i,1-t_i} := \mathsf{PKE.Enc}\left(\mathsf{pke.pk}_{i,1-t_i}, 0^{\ell_{\mathrm{op}}+\ell_{\mathsf{vk}}+1}; \tilde{r}_i\right)$$
3. Compute the PRC ciphertexts and the header.
   (a) For each $i \in [n]$ and $b \in \{0,1\}$, $c_{i,b} := \mathsf{PRC.Enc}(\mathsf{prc.pk}_{i,b}, (\mathsf{pke.ct}_{i,b}, \mathsf{com}_i); \mathsf{PRF}(\mathsf{K}, (i,b)))$.
   (b) Set $c := \mathsf{PRG}(r_{00}) + \mathsf{ECC.Encode}(m, \mathsf{K}, (\mathsf{pke.ct}_{i,b}, \mathsf{com}_i)_{i,b}) \cdot B_{\mathsf{pl}}$.
4. Computing the digital signature:
   (a) Let $\mathsf{ds}.x = \left(c, (c_{i,0}, c_{i,1})_{i\in[n]}\right)$. Compute $\mathsf{ds}.\sigma \leftarrow \mathsf{DS.Sign}(\mathsf{ds.sk}, \mathsf{ds}.x)$. Pad the signature so that the length is ℓ_{PRG}.
   (b) Compute $\sigma = \mathsf{PRG}(r_{01}) \oplus \mathsf{ECC.Encode}\left(\mathsf{ecc.pp}, \mathsf{ds}.\sigma\right)$.
5. Output $\mathsf{ct} = (\mathsf{ds}.x, \sigma)$.

[10] If needed, the message is padded so that the PRC codeword length is $\ell_{\mathrm{PRG}}/2n$.

Note that the final ciphertext consists of three parts, each of length ℓ_{PRG}: c, $(c_{i,b})_{i,b}$ and σ.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$**:** Parse the secret key $\mathsf{sk} := (\mathsf{pke.sk}_{i,0}, \mathsf{prc.sk}_{i,0})_{i\in[n]}$ and the ciphertext $\mathsf{ct} = (\mathsf{ds}.x, \sigma)$ where $\mathsf{ds}.x = \left(c, \{c_{i,0}, c_{i,1}\}_{i\in[n]}\right)$:

1. Compute $t = \mathsf{PRC.Find}\left(\mathsf{sk}, (c_{i,0})_i\right)$ (Figure 1).
2. Compute $\left(\mathsf{msg}, \mathsf{K}, c', \left(c'_{i,b}, \mathsf{pke.ct}_{i,b}, \mathsf{com}_i\right)_{i,b}, \mathsf{ds}.\sigma\right) \leftarrow \mathsf{PRC.CheckNoise}\left(c, (c_{i,b})_{i,b}, \sigma, t, (\mathsf{prc.sk}_{i,0})_i\right)$ (Figure 2).
3. Let $\mathsf{ds}.x' = \left(c', \left(c'_{i,b}\right)_{i,b}\right)$. Compute $\mathsf{ds.vk} \leftarrow \mathsf{PRC.CheckPKE}(t, (\mathsf{pke.ct}_{i,t_i}, \mathsf{com}_i)_i)$ (Figure 3).
4. If $\mathsf{DS.Verify}\left(\mathsf{ds.vk}, \mathsf{ds}.x', \mathsf{ds}.\sigma\right) = 1$, output msg, else output $\perp$.

PRC.Find

Inputs:

Secret Key $\mathsf{sk} = (\mathsf{pke.sk}_{i,0}, \mathsf{prc.sk}_{i,0})_{i\in[n]}$
PRC ciphertexts $(c_{i,0})_{i\in[n]}$

Output: t

1. For each $i \in [n]$, do the following:
   (a) Let $(\underline{\mathsf{pke.ct}_{i,0}}, \underline{\mathsf{com}_i}) = \mathsf{PRC.Decode}(\mathsf{prc.sk}_{i,0}, c_{i,0})$
   (b) Let $\underline{m_i} = \mathsf{PKE.Dec}(\mathsf{pke.sk}_{i,0}, \underline{\mathsf{pke.ct}_{i,0}})$.
   (c) If $\underline{m_i} = 1 \,\|\, \underline{\mathsf{op}_i} \,\|\, \underline{\mathsf{ds.vk}_i}$ and $\mathsf{Open}(\mathsf{com.pp}, 0, \underline{\mathsf{com}_i}, \underline{\mathsf{op}_i}, \underline{\mathsf{ds.vk}_i}) = \top$, set $\tilde{t}_i = 0$. Else set $\tilde{t}_i = 1$.
2. Compute $s = \mathsf{ECC.Decode}\left(\mathsf{ecc.pp}, \tilde{t}\right)$ and $t = \mathsf{ECC.Encode}\left(\mathsf{ecc.pp}, s\right)$.
3. Output t.

Fig. 1. Routine PRC.Find.

Information rate: Suppose that ECC encodes t-bit messages into codewords of length $k_{\mathrm{ecc}} \cdot t$ for some constant $k_{\mathrm{ecc}} > 1$, and let ℓ_{msg} denote the message length in our PRC scheme. Since both $|\mathsf{op}_i|$ and $|\mathsf{ds.vk}|$ are polynomial in the security parameter, it follows that $|\mathsf{pke.ct}_{i,t_i}| = \mathsf{poly}(\lambda)$. Likewise, because $|\mathsf{com}_i| = \mathsf{poly}(\lambda)$, we have $|c_{i,b}| = \mathsf{poly}(\lambda)$. As $n = \mathsf{poly}(\lambda)$, this further implies that $|\{c_{i,b}\}| = \mathsf{poly}(\lambda)$. Since, $|\mathsf{K}| = \mathsf{poly}(\lambda)$, total payload size is therefore $|c| = k_{\mathrm{ecc}} \cdot (\ell_{\mathsf{msg}} + \mathsf{poly}(\lambda))$, while the signature contributes $O(\lambda)$ additional bits. Consequently, the information rate of the scheme is

$$\frac{\ell_{\mathsf{msg}}}{k_{\mathrm{ecc}} \cdot (\ell_{\mathsf{msg}} + \mathsf{poly}(\lambda)) + \mathsf{poly}(\lambda)} = \left(k_{\mathrm{ecc}} + \frac{k_{\mathrm{ecc}} \cdot \mathsf{poly}(\lambda)}{\ell_{\mathsf{msg}}} + \frac{\mathsf{poly}(\lambda)}{\ell_{\mathsf{msg}}}\right)^{-1}.$$

In particular, as ℓ_{msg} grows large, the information rate approaches k_{ecc}^{-1}.

Theorem 8 (Adaptive Robustness). *Assuming* PRC *is* α*-robust,* ECC *is* γ*-robust, then the above scheme is* $\frac{\alpha\cdot\gamma}{6}$*- adaptively robust.*

Proof. Let $\ell_c := |c|, \ell_{\mathsf{cw}} = |c_{i,b}|, \ell_\sigma = |\sigma|$, we have $|\ell_c| = |\ell_\sigma| = 2n \cdot \ell_{\mathsf{cw}}$, and $\ell_{\mathsf{ct}} := \ell_c + 2n \cdot \ell_{\mathsf{cw}} + \ell_\sigma$. Let $\delta = \alpha\gamma/6$. We then need to argue, to prove adaptive corruptness, that an adversary cannot output a tuple (m, r, x) where $\mathsf{Ham}(\mathsf{Enc}(\mathsf{pk}, m; r), x) \le \delta\ell_{\mathsf{ct}}$ while $m \neq \mathsf{Dec}(\mathsf{sk}, x)$ with non-negligible probability. To prove this, we argue something stronger, we consider an adversary that submits a message-randomness pair (m, r)

PRC.CheckNoise

Inputs:

$c, (c_{i,b})_{i,b}, \sigma$ (the ciphertext)
t (output by Find)
$(\mathsf{prc.sk}_{i,0})_i$

Output: $\left(\mathsf{msg}, \mathsf{K}, c', (c'_{i,b}, \mathsf{pke.ct}_{i,b}, \mathsf{com}_i)_{i,b}, \mathsf{ds}.\sigma\right)$

1. Computing noise-free version of ciphertext:
   (a) Compute $(r_{00}, r_{01}, r_1, \ldots, r_n) = \mathsf{HPRG.Eval}(\mathsf{hprg.pp}, t)$.
   (b) Compute $\left(\mathsf{msg}, \mathsf{K}, (\mathsf{pke.ct}_{i,b}, \mathsf{com}_i)_{i,b}\right) = \mathsf{ECC.Decode}(\mathsf{ecc.pp}, \left((c - \mathsf{PRG}(r_{00})) \cdot B_{\mathsf{pl}}^{-1}\right))$.
   (c) Compute $\mathsf{ds}.\sigma = \mathsf{ECC.Decode}(\mathsf{ecc.pp}, (\mathsf{PRG}(r_{01}) \oplus \sigma))$.
   (d) Compute $c' = \mathsf{PRG}(r_{00}) + \mathsf{ECC.Encode}(\mathsf{ecc.pp}, ((\mathsf{msg}, \mathsf{K}, (\mathsf{pke.ct}_{i,b}, \mathsf{com}_i)_{i,b}))) \cdot B_{\mathsf{pl}}$.
   (e) For each i, b, compute $c'_{i,b} = \mathsf{PRC.Encode}(\mathsf{prc.pk}_{i,b}, \{\mathsf{pke.ct}_{i,b}, \mathsf{com}_i\}_{i,b}; \mathsf{PRF.Eval}(\mathsf{K}, (i, b)))$.
   (f) Compute $\sigma' = \mathsf{PRG}(r_{01}) \oplus \mathsf{ECC.Encode}(\mathsf{ecc.pp}, \mathsf{ds}.\sigma)$.
2. Checking noise is within bounds:
   (a) If $\Delta\left(\left(c, (c_{i,b})_{i,b}, \sigma\right), \left(c', (c'_{i,b})_{i,b}, \sigma'\right)\right) > \delta\ell_{\mathsf{ct}}$, output $\perp$.
3. Checking the commitments and ciphertexts in c are consistent with the commitments and ciphertexts used in Find.
   (a) For each $i \in [n]$, compute $\left(\underline{\mathsf{pke.ct}_{i,0}}, \underline{\mathsf{com}_i}\right) = \mathsf{PRC.Decode}(\mathsf{prc.sk}_{i,0}, c_{i,0})$.
   (b) For each $i \in [n]$ s.t. $\Delta(c_{i,b}, c'_{i,b}) \leq \alpha\ell_{\mathsf{cw}}$ for both $b \in \{0,1\}$, check $\mathsf{com}_i = \underline{\mathsf{com}_i}$ and $\mathsf{pke.ct}_{i,0} = \underline{\mathsf{pke.ct}_{i,0}}$. If any checks fail, output $\perp$.
4. Output $\left(\mathsf{msg}, \mathsf{K}, c', (c'_{i,b}, \mathsf{pke.ct}_{i,b}, \mathsf{com}_i)_{i,b}, \mathsf{ds}.\sigma\right)$.

Fig. 2. Routine PRC.CheckNoise.

PRC.CheckPKE

Inputs:

$(\mathsf{pke.pk}_{i,b})_{i,b}$ (part of public key)
t (produced by Find).
$(\mathsf{pke.ct}_{i,t_i}, \mathsf{com}_i)_{i,b}$ (commitments and ciphertexts produced by CheckNoise)

Output: ds.vk

1. Compute $(r_{00}, r_{01}, r_1, \ldots, r_n) = \mathsf{HPRG.Eval}(\mathsf{hprg.pp}, t)$.
2. For each $i \in [n]$, do the following:
   (a) If $t_i = 0$, compute $m_i = \mathsf{PKE.Recover}(\mathsf{pke.pk}_{i,0}, \mathsf{pke.ct}_{i,0}, r_i)$. Perform the following checks. If any of the checks fail, output $\perp$.
      i. $m_i = 1 \| \mathsf{op}_i \| \mathsf{ds.vk}_i$ and $\mathsf{Open}(\mathsf{com.pp}, 0, \mathsf{com}_i, \mathsf{op}_i, \mathsf{ds.vk}_i) = \top$.
      ii. $\mathsf{pke.ct}_{i,0} = \mathsf{PKE.Enc}(\mathsf{pke.pk}_{i,0}, m_i; r_i)$.
   (b) If $t_i = 1$, compute $m_i = \mathsf{PKE.Recover}(\mathsf{pke.pk}_{i,1}, \mathsf{pke.ct}_{i,1}, r_i)$. Perform the following checks. If any of the checks fail, output $\perp$.
      i. $m_i = 1 \| \mathsf{op}_i \| \mathsf{ds.vk}_i$ and $\mathsf{Open}(\mathsf{com.pp}, 1, \mathsf{com}_i, \mathsf{op}_i, \mathsf{ds.vk}_i) = \top$.
      ii. $\mathsf{pke.ct}_{i,1} = \mathsf{PKE.Enc}(\mathsf{pke.pk}_{i,1}, m_i; r_i)$.
3. If $\mathsf{ds.vk}_i = \mathsf{ds.vk}$ for all $i \in [n]$, output ds.vk. Else output $\perp$.

Fig. 3. Routine PRC.CheckPKE.

as well as three error vectors $\mathbf{e}_1 \in \{0,1\}^{\ell_c}, \mathbf{e}_2 \in \{0,1\}^{2n\ell_{\mathsf{cw}}}, \mathbf{e}_3 \in \{0,1\}^{\ell_\sigma}$ with $\mathsf{Ham}(\mathbf{e}_i) \leq \delta\ell_{\mathsf{ct}}$ for all $i \in [3]$.

Now, we say that such an adversary wins the game if it holds that

$$\mathsf{Dec}\left(\mathsf{sk}, \left(c, (c_{i,b})_{i\in[n],b\in\{0,1\}}, \sigma\right)\right) \neq \mathsf{Dec}\left(\mathsf{sk}, \left(c \oplus \mathbf{e}_1, (c_{i,b})_{i\in[n],b\in\{0,1\}} \oplus \mathbf{e}_2, \sigma \oplus \mathbf{e}_3\right)\right)$$

with $\mathsf{ct} = \left(c, (c_{i,b})_{i,b}, \sigma\right) = \mathsf{Enc}(\mathsf{pk}, m; r)$. We can directly see that such a game achieves stronger security than the adaptive robustness game since the adaptive robustness adversary is only allowed to submit a single vector $\mathbf{e} \in \{0,1\}^{\ell_c + 2n\ell_{\mathsf{cw}} + \ell_\sigma}$ with $\mathsf{Ham}(\mathbf{e}) \leq \delta\ell_{\mathsf{ct}}$. Let $c' \coloneqq c \oplus \mathbf{e}_1$, $c'_{i,b} \coloneqq c_{i,b} \oplus \mathbf{e}_2$, $\sigma' \coloneqq \sigma \oplus \mathbf{e}_3$, and let $\mathsf{ct}' = \left(c', \left(c'_{i,b}\right)_{i,b}, \sigma'\right)$.
To prove the above equality, it suffices to show that the procedures Find and $\mathsf{CheckNoise}$ output the same for the ciphertexts ct and ct'. Now, we are walking through the different steps of the two procedures to bound these probabilities.

Claim. $\mathsf{Find}(\mathsf{sk}, \mathsf{ct}) = \mathsf{Find}(\mathsf{sk}, \mathsf{ct}')$.

Proof. First, note that if t is the HPRG seed used to encrypt m, then Step 1 of $\mathsf{Find}(\mathsf{sk}, \mathsf{ct})$ computes $\tilde{t} = t$, and as a result, $\mathsf{Find}(\mathsf{sk}, \mathsf{ct})$ outputs t. This follows from the perfect correctness of the PKE and commitment schemes. We will now argue that if $\mathsf{Ham}(\mathbf{e}_2) \leq \delta \cdot \ell_{\mathsf{ct}}$, then Step 1 of $\mathsf{Find}(\mathsf{sk}, \mathsf{ct}')$ produces a $\tilde{t'}$ such that $\Delta\left(\tilde{t'}, t\right) \leq \gamma \cdot n$, assuming α-adaptive robustness of the base PRC scheme.
Parse $\mathbf{e}_2 = (\mathbf{e}_{2,i,b})_{i\in[n],b\in\{0,1\}}$, and let $\mathcal{I} = \{i : \mathsf{Ham}\left(\mathbf{e}_{2,i,b}\right) \leq \alpha \cdot \ell_{\mathsf{cw}} \text{ for both } b \in \{0,1\}\}$. Since $\delta = \alpha \cdot \gamma/6$, it follows that $|\mathcal{I}| \geq (1-\gamma) \cdot n$.[11] Using the adaptive robustness of the base PRC scheme (and the perfect correctness of the PKE scheme and the commitment scheme), we get that $\tilde{t'_i} = t_i$ for all $i \in \mathcal{I}$. In particular, if $c_{i,b}$ is the PRC encoding of $(\mathsf{pke.ct}_{i,b}, \mathsf{com}_i)$ with randomness $z_{i,b}$ and $\mathsf{Dec}\left(\mathsf{prc.sk}_{i,b}, (c_{i,b} \oplus \mathbf{e}_{2,i,b})\right) \neq (\mathsf{pke.ct}_{i,b}, \mathsf{com}_i)$, then the reduction algorithm can break α-adaptive robustness of the base PRC scheme by sending $(\mathsf{pke.ct}_{i,b}, \mathsf{com}_i)$ as the message, $z_{i,b}$ as the randomness and $(c_{i,b} \oplus \mathbf{e}_{2,i,b})$ as the codeword. □

It remains to analyze the behavior of $\mathsf{CheckNoise}$ on the inputs ct and ct' where both procedures take as an input the same t (which follows from the above analysis of Find). Given that both procedures take as an input the same t, it follows that step 1 (a) will also output the same values. Next, we rely on the robustness of ECC. In more detail, since $\delta\ell_{\mathsf{ct}} \leq \gamma\ell_c = \gamma\ell_\sigma$, for the steps 1. (b) and (c), it must hold that

$$\mathsf{ECC.Decode}\left(\mathsf{ecc.pp}, \left(c - \mathsf{PRG}\left(r_{00}\right) \cdot B_{\mathsf{pl}}^{-1}\right)\right) = \mathsf{ECC.Decode}\left(\mathsf{ecc.pp}, \left((c \oplus \mathbf{e}_1) - \mathsf{PRG}\left(r_{00}\right) \cdot B_{\mathsf{pl}}^{-1}\right)\right)$$
$$\mathsf{ECC.Decode}\left(\mathsf{ecc.pp}, (\mathsf{PRG}\left(r_{01}\right) \oplus \sigma)\right) = \mathsf{ECC.Decode}(\mathsf{ecc.pp}, (\mathsf{PRG}\left(r_{01}\right) \oplus \sigma \oplus \mathbf{e}_3)),$$

If the above holds, then the steps 1 (d)-(f) directly have the same output as well. The bounds in step 2. are all satisfied due to our restriction $\mathsf{Ham}(\mathbf{e}_i) \leq \delta\ell_{\mathsf{ct}}$. To argue that step 3 yields the same output, we can, again, rely on the adaptive robustness as it has been done in the beginning of the proof. In more detail, the decoding output computed in step 3 (a) is the same as the decoding that is computed as the 1. step of Find. Only violating adaptive robustness would lead to a failing check in the step 3 (b), given the above analysis involving the error-correcting codes. □

Theorem 9. *Assuming that* PKE *is an IND-CPA secure PKE scheme with almost perfect-correctness,* PRC *is an α adaptively robust IND-CPA secure PRC scheme,* ECC *is a γ-robust error-correcting code,* Com *is a commitment scheme satisfies statistical binding, equivocality, pseudorandomness of commitments and independence of tag for 0-commitments,* HPRG *is a secure hinting PRG,* PRG *is a secure PRG,* PRF *is a secure PRF and* DS *is a secure signature scheme. Then the above construction is a IND-CCA secure PRC scheme.*

Proof. Our proof proceeds via a sequence of hybrids. First, we will describe all hybrids, and then show that the hybrids are computationally indistinguishable.

Hybrid $\mathcal{H}_0$: The corresponds to the original security game with a few modifications as follows.

[11] If $|\mathcal{I}| < (1-\gamma)\cdot n$, then there exist at least $\gamma \cdot n$ indices i where either $\mathsf{Ham}\left(\mathbf{e}_{2,i,0}\right) > \alpha \cdot \ell_{\mathsf{cw}}$ or $\mathsf{Ham}\left(\mathbf{e}_{2,i,1}\right) > \alpha \cdot \ell_{\mathsf{cw}}$. Therefore, $\mathsf{Ham}(\mathbf{e}) > \gamma \cdot n \cdot (\alpha \cdot \ell_{\mathsf{cw}})$. But this is a contradiction, since we assumed $\mathsf{Ham}(\mathbf{e}) \leq \delta \cdot \ell_{\mathsf{ct}}$, $\delta = \alpha \cdot \gamma/6$ and $\ell_{\mathsf{ct}} = 6n \cdot \ell_{\mathsf{cw}}$

Description of Hybrid $\mathcal{H}_0$

- **Initialization Phase:** The challenger performs the following steps:
  1. Sample $\mathsf{ecc.pp} \leftarrow \mathsf{ECC.Setup}(1^\lambda)$.
  2. Sample $\mathsf{hprg.pp} \leftarrow \mathsf{HPRG.Setup}(1^\lambda)$
  3. Sample $2n$ different PKE keys, i.e., $(\mathsf{pke.pk}_{i,b}, \mathsf{pke.sk}_{,i,b}) \leftarrow \mathsf{PKE.Setup}(1^\lambda)$ for all $b \in \{0,1\}, i \in [n]$.
  4. Sample $2n$ different PRC keys, i.e., $(\mathsf{prc.pk}_{i,b}, \mathsf{prc.sk}_{i,b}) \leftarrow \mathsf{PRC.Setup}(1^\lambda)$ for all $b \in \{0,1\}, i \in [n]$.
  5. Sample $\mathsf{com.pp} \leftarrow \mathsf{Com.Setup}\left(1^\lambda, 1^n\right)$.
  6. Sample $B_{\mathsf{pl}} \leftarrow \{0,1\}^{\ell_{\mathrm{PRG}}}$.
  7. It sets $\mathsf{pk} = \left(\mathsf{ecc.pp}, \mathsf{hprg.pp}, B_{\mathsf{pl}}, \mathsf{com.pp}, \left(\mathsf{pke.pk}_{i,b}, \mathsf{prc.pk}_{i,b}\right)_{i\in[n],b\in\{0,1\}}\right)$ and $\mathsf{sk} = (\mathsf{pke.sk}_{i,0}, \mathsf{prc.sk}_{i,0})_{i\in[n]}$.
  8. It sends pk to the adversary.
  9. Sample $s^* \leftarrow \{0,1\}^\lambda$.
  10. Set $t^* = \mathsf{ECC.Encode}(\mathsf{ecc.pp}, s^*)$.
  11. Sample $\mathsf{K}^* \leftarrow \mathsf{PRF.Setup}(1^\lambda)$.
  12. Sample $(\mathsf{ds.vk}^*, \mathsf{ds.sk}^*) \leftarrow \mathsf{DS.Setup}(1^\lambda)$.
  13. Compute $(\mathsf{com}_i^*, \mathsf{op}_i^*)_i \leftarrow \mathsf{Commit}(\mathsf{com.pp}, t^*, \mathsf{ds.vk}^*)$.
  14. Let $(r_{00}^*, r_{01}^*, r_1^*, \ldots, r_n^*) = \mathsf{HPRG.Eval}(\mathsf{hprg.pp}, t^*)$. For each $i \in [n]$, choose $\tilde{r}_i^* \leftarrow \{0,1\}^\ell$.
  15. For each $i \in [n]$,
      - Compute $\mathsf{pke.ct}_{i,t_i^*}^* := \mathsf{PKE.Enc}\left(\mathsf{pke.pk}_{i,t_i^*}, 1\|\mathsf{op}_i^*\|\mathsf{ds.vk}^*; r_i^*\right)$
      - Compute $\mathsf{pke.ct}_{i,1-t_i^*}^* := \mathsf{PKE.Enc}\left(\mathsf{pke.pk}_{i,1-t_i^*}, 0^{\ell_{\mathrm{op}}+\ell_{\mathsf{vk}}+1}; \tilde{r}_i^*\right)$
      - Compute $c_{i,b}^* := \mathsf{PRC.Enc}(\mathsf{prc.pk}_{i,b}, (\mathsf{pke.ct}_{i,b}^*, \mathsf{com}_i^*); \mathsf{PRF}(\mathsf{K}^*, (i,b)))$.

- **Pre-Challenge Phase:** The adversary can adaptively query the decryption oracle polynomially many times. For each query $\mathsf{ct} = (\mathsf{ds}.x, \sigma)$ where $\mathsf{ds}.x = \left(c, (c_{i,0}, c_{i,1})_i\right)$, the challenger computes
  - $t = \mathsf{PRC.Find}\left(\mathsf{sk}, (c_{i,0})_i\right)$,
  - $\left(\mathsf{msg}, \mathsf{K}, (\mathsf{pke.ct}_{i,b}, \mathsf{com}_i)_i, \mathsf{ds}.\sigma\right) \leftarrow \mathsf{CheckNoise}\left(c, (c_{i,b})_{i,b}, \sigma, t\right)$,
  - $\mathsf{ds.vk} \leftarrow \mathsf{CheckPKE}\left(t, (\mathsf{pke.ct}_{i,t_i}, \mathsf{com}_i)\right)$,
  - outputs msg if $\mathsf{DS.Verify}\left(\mathsf{ds.vk}, \mathsf{ds}.x, \mathsf{ds}.\sigma\right) = 1$.

- **Challenge phase:** The adversary sends m^* to the challenger who performs the following steps (recall, some of the components used in this phase were already computed in the Setup Phase).
  1. It sets $c^* := \mathsf{PRG}(r_{00}^*) + \mathsf{ECC.Encode}(\mathsf{ecc.pp}, (m^*, \mathsf{K}^*, (\mathsf{pke.ct}_{i,b}^*, \mathsf{com}_i^*)_{i\in[n],b\in\{0,1\}})) \cdot B_{\mathsf{pl}}$.
  2. It sets $\mathsf{ds}.x^* = \left(c^*, \left(c_{i,0}^*, c_{i,1}^*\right)_{i\in[n]}\right)$.
  3. It computes $\mathsf{ds}.\sigma^* \leftarrow \mathsf{DS.Sign}\left(\mathsf{ds.sk}^*, \mathsf{ds}.x^*\right)$.
  4. It compute $\sigma^* = \mathsf{PRG}\left(r_{01}^*\right) \oplus \mathsf{ECC.Encode}(\mathsf{ecc.pp}, \mathsf{ds}.\sigma^*)$.
  5. It returns $\mathsf{ct}^* = (\mathsf{ds}.x^*, \sigma^*)$.

- **Post-Challenge Phase:** The adversary can adaptively query the decryption oracle polynomially many times. These queries are handled similar to the pre-challenge decryption queries.
- **Response Phase:** The adversary outputs a bit β.

Hybrid $\mathcal{H}_1$: This hybrid proceeds as the previous one, with the only difference that the decryption oracle rejects queries where the underlying verification key $\mathsf{ds.vk}$ is equal to the verification key $\mathsf{ds.vk}^*$ used in the challenge encryption query. The indistinguishability of this and the previous hybrid can be argued using the unforgeability of the signature scheme DS by showing that the probability of an adversary submitting such a query is negligible. The details on this are given in Lemma 6.

Hybrid $\mathcal{H}_2$: In this hybrid, during the initialization phase, the commitment scheme is in equivocal mode, i.e., $\left(\mathsf{com.pp}, \left(\mathsf{com}_i^*, \mathsf{op}_{i,0}^*, \mathsf{op}_{i,1}^*\right)_i\right) \leftarrow \mathsf{Equiv}(1^\lambda, 1^n, \mathsf{ds.vk}^*)$ and $\mathsf{op}_i^* := \mathsf{op}_{i,t_i^*}^*$. The indistinguishability of this and the previous hybrid can be argued using the equivocality of Com. The details on this are given in Lemma 7.

Hybrid $\mathcal{H}_3$: In this hybrid, the decryption queries are handled using the procedures $\mathsf{PRC.Find}^\dagger$ (see Fig. 4) and $\mathsf{PRC.CheckNoise}^\dagger$ (see Fig. 5) on path t^*, instead of $\mathsf{PRC.Find}$ and $\mathsf{PRC.CheckNoise}$. To argue the indistinguishability between this and the previous hybrid, we rely on the perfect correctness of PKE and ECC, and the adaptive robustness of PRC. This transition is quite delicate, and we prove it in detail in Lemma 8.

$\mathsf{PRC.Find}^\dagger$

Inputs:

1. path $\mathfrak{p} \in \{0,1\}^n$.
2. secret keys $(\mathsf{pke.sk}_{i,\mathfrak{p}_i}, \mathsf{prc.sk}_{i,\mathfrak{p}_i})_{i\in[n]}$
3. Partial Ciphertext $(c_{i,0}, c_{i,1})_{i\in[n]}$

Output: $t^\dagger \in \{0,1\}^n$

1. For each $i \in [n]$, do the following:
   (a) If $\mathfrak{p}_i = 0$, then
      i. Let $(\underline{\mathsf{pke.ct}^\dagger_{i,0}}, \underline{\mathsf{com}^\dagger_i}) = \mathsf{PRC.Decode}(\mathsf{prc.sk}_{i,0}, c_{i,0})$.
      ii. Let $\underline{m^\dagger_i} = \mathsf{PKE.Dec}(\mathsf{pke.sk}_{i,0}, \underline{\mathsf{pke.ct}^\dagger_{i,0}})$.
      iii. If $\underline{m^\dagger_i} = 1 \,\|\, \underline{\mathsf{op}^\dagger_i} \,\|\, \underline{\mathsf{ds.vk}^\dagger_i}$ and $\mathsf{Open}\left(\mathsf{com.pp}, 0, \underline{\mathsf{com}^\dagger_i}, \underline{\mathsf{op}^\dagger_i}\right) = \top$, set $\tilde{t}^\dagger_i = 0$. Else set $\tilde{t}^\dagger_i = 1$.
   (b) If $\mathfrak{p}_i = 1$, then
      i. Let $(\underline{\mathsf{pke.ct}^\dagger_{i,1}}, \underline{\mathsf{com}^\dagger_i}) = \mathsf{PRC.Decode}(\mathsf{prc.sk}_{i,1}, c_{i,1})$.
      ii. Let $\underline{m^\dagger_i} = \mathsf{PKE.Dec}(\mathsf{pke.sk}_{i,1}, \underline{\mathsf{pke.ct}^\dagger}_{i,1})$.
      iii. If $\underline{m^\dagger_i} = 1 \,\|\, \underline{\mathsf{op}^\dagger_i} \,\|\, \underline{\mathsf{ds.vk}^\dagger_i}$ and $\mathsf{Open}\left(\mathsf{com.pp}, 1, \underline{\mathsf{com}^\dagger_i}, \underline{\mathsf{op}^\dagger_i}, \underline{\mathsf{ds.vk}^\dagger_i}\right) = \top$, set $\tilde{t}^\dagger_i = 1$. Else set $\tilde{t}^\dagger_i = 0$.
2. Compute $s^\dagger = \mathsf{ECC.Decode}(\mathsf{ecc.pp}, \tilde{t}^\dagger)$ and $t^\dagger = \mathsf{ECC.Encode}(\mathsf{ecc.pp}, s^\dagger)$.
3. Output $t^\dagger$.

Fig. 4. Routine $\mathsf{PRC.Find}^\dagger$.

Hybrid $\mathcal{H}_4$: In this hybrid, the ciphertexts $\mathsf{pke.ct}^*_{i,\overline{t^*_i}}$ are generated as $\mathsf{PKE.Enc}(\mathsf{pke}_{i,\overline{t^*_i}}, 1\|\mathsf{op}_{i,\overline{t^*_i}}\|\mathsf{ds.vk}^*; r^*_i)$ for all $i \in [n]$. The indistinguishability of this and the previous hybrid can be argued using the IND-CPA security of PKE. The details on this are given in Lemma 13.
Hybrid $\mathcal{H}_5$: In this hybrid, the procedure $\mathsf{PRC.Find}$ instead of $\mathsf{PRC.Find}^\dagger$ is executed in the pre- and post-challenge phases. The transition from the previous hybrid to this hybrid proceeds analogously to the transition from $\mathcal{H}_2$ to $\mathcal{H}_3$, therefore, we refer to Lemma 8 for more details.
Hybrid $\mathcal{H}_6$: In this hybrid, the values $\tilde{r}^*_i, c^*$ and σ^* are chosen uniformly at random for all $i \in [n]$. The indistinguishability of this and the previous hybrid can be argued using the security of the hinting PRG HPRG and the standard PRG PRG. The details on this are given in Lemma 14.
Hybrid $\mathcal{H}_7$: In this hybrid, the ciphertexts $c^*_{i,b}$ for all $i \in [n], b \in \{0,1\}$ are generated as $c^*_{i,b} \coloneqq \mathsf{PRC.Enc}(\mathsf{prc.pk}_{i,b}, (\mathsf{pke.ct}^*_{i,b}, \mathsf{com}^*_i); u_{i,b})$ where $u_{i,b}$ are sampled uniformly at random. The indistinguishability of this and the previous hybrid can be argued using the pseudorandomness of PRF. The details on this are given in Lemma 15.
Hybrid $\mathcal{H}_8$: In this hybrid, the challenger chooses $c^*_{i,1}$ uniformly at random for all $i \in [n]$. The indistinguishability of this and the previous hybrid can be argued using the pseudorandomness of PRC. The details on this are given in Lemma 16.
Hybrid $\mathcal{H}_9$: In this hybrid, the challenger uses $\mathsf{PRC.Find}^\dagger$ on path $\overline{1}$ instead of $\mathsf{PRC.Find}$ in the pre-challenge and post-challenge phases. The transition from the previous hybrid to this hybrid proceeds analogously to the transition from $\mathcal{H}_2$ to $\mathcal{H}_3$, therefore, we refer to Lemma 8 for more details.
Hybrid $\mathcal{H}_{10}$: In this hybrid, the challenger chooses $c^*_{i,0}$ uniformly at random for all $i \in [n]$. The transition from the previous hybrid to this hybrid proceeds analogously to the transition from $\mathcal{H}_7$ to $\mathcal{H}_8$, relying on the pseudorandomness of PRC. We refer to Lemma 16 for more details.

PRC.CheckNoise†

Inputs:

$c, (c_{i,b})_{i,b}, \sigma$ (the ciphertext)
$t^\dagger$ (output by $\mathsf{Find}^\dagger$)
path $\mathfrak{p} \in \{0,1\}^n$

Output: $\left(\mathsf{msg}^\dagger, \mathsf{K}^\dagger, \left(\mathsf{pke.ct}^\dagger_{i,b}, \mathsf{com}^\dagger_i\right)_{i,b}, \mathsf{ds}.\sigma^\dagger\right)$

1. Computing noise-free version of ciphertext:
   (a) Compute $(r_{00}, r_{01}, r_1, \ldots, r_n) = \mathsf{HPRG.Eval}\left(\mathsf{hprg.pp}, t^\dagger\right)$.
   (b) Compute $\left(\mathsf{msg}^\dagger, \mathsf{K}^\dagger, \left(\mathsf{pke.ct}^\dagger{}_{i,b}, \mathsf{com}^\dagger{}_i\right)_{i,b}\right) = \mathsf{ECC.Decode}(\mathsf{ecc.pp}, \left((c - \mathsf{PRG}\left(r_{00}\right)) \cdot B_{\mathsf{pl}}^{-1}\right))$.
   (c) Compute $\mathsf{ds}.\sigma^\dagger = \mathsf{ECC.Decode}\left(\mathsf{ecc.pp}, (\mathsf{PRG}\left(r_{01}\right) \oplus \sigma)\right)$.
   (d) Compute $c^{\dagger'} = \mathsf{ECC.Encode}(\mathsf{ecc.pp}, (\mathsf{PRG}\left(r_{00}\right) + (\mathsf{msg}^\dagger, \mathsf{K}^\dagger, (\mathsf{pke.ct}^\dagger{}_{i,b}, \mathsf{com}^\dagger{}_i)_{i,b}))) \cdot B_{\mathsf{pl}}$.
   (e) For each i, b, compute $c^{\dagger'}_{i,b} = \mathsf{PRC.Encode}(\mathsf{prc.pk}_{i,b}, (\mathsf{pke.ct}^\dagger{}_{i,b}, \mathsf{com}^\dagger{}_i)_{i,b}; \mathsf{PRF.Eval}(\mathsf{K}^\dagger, (i,b)))$.
   (f) Compute $\sigma^{\dagger'} = \mathsf{ECC.Encode}\left(\mathsf{ecc.pp}, \mathsf{ds}.\sigma^\dagger\right)$.
2. Checking noise is within bounds:
   (a) If $\Delta\left(\left(c, (c_{i,b})_{i,b}, \sigma\right), \left(c', (c'_{i,b})_{i,b}, \sigma'\right)\right) > \delta\ell_{\mathsf{ct}}$, output $\perp$.
3. Checking the commitments and ciphertexts in c are consistent with the commitments and ciphertexts used in Find.
   (a) For each $i \in [n]$, compute $\left(\underline{\mathsf{pke.ct}}^\dagger{}_{i,\mathfrak{p}_i}, \underline{\mathsf{com}^\dagger_i}\right) = \mathsf{PRC.Decode}(\mathsf{prc.sk}_{i,\mathfrak{p}_i}, c_{i,\mathfrak{p}_i})$.
   (b) For each $i \in [n]$ s.t. $\Delta\left(c^\dagger_{i,b}, c^{\dagger'}_{i,b}\right) \leq \alpha\ell_{\mathsf{cw}}$ for both $b \in \{0,1\}$, check $\mathsf{com}^\dagger{}_i = \underline{\mathsf{com}^\dagger_i}$ and $\mathsf{pke.ct}^\dagger{}_{i,\mathfrak{p}_i} = \underline{\mathsf{pke.ct}}^\dagger{}_{i,\mathfrak{p}_i}$. If any checks fail, output $\perp$.
4. Output $\left(\mathsf{msg}^\dagger, \mathsf{K}^\dagger, \left(\mathsf{pke.ct}^\dagger{}_{i,b}, \mathsf{com}^\dagger{}_i\right)_{i,b}, \mathsf{ds}.\sigma^\dagger\right)$.

Fig. 5. Routine $\mathsf{PRC.CheckNoise}^\dagger$.

Hybrid $\mathcal{H}_{11}$: In this hybrid, the challenger uses $\mathsf{PRC.Find}$ instead of $\mathsf{PRC.Find}^\dagger$ in the pre-challenge and post-challenge phases. The transition from the previous hybrid to this hybrid proceeds analogously to the transition from $\mathcal{H}_2$ to $\mathcal{H}_3$, therefore, we refer to Lemma 8 for more details.
Hybrid $\mathcal{H}_{12}$: In this hybrid, during the initialization phase, the setup of the commitment scheme is changed back to the standard setup, i.e., $\mathsf{com.pp} \leftarrow \mathsf{Com.Setup}(1^\lambda, 1^n)$. The transition from the previous hybrid to this hybrid proceeds analogously to the transition from $\mathcal{H}_1$ to $\mathcal{H}_2$, relying on the equivocality of Com. We refer to Lemma 7 for more details.
Hybrid $\mathcal{H}_{13}$: In this hybrid, during the decryption queries, the challenger does not reject queries when $\mathsf{ds.vk} = \mathsf{ds.vk}^*$. The transition from the previous hybrid to this hybrid proceeds analogously to the transition from $\mathcal{H}_0$ to $\mathcal{H}_1$, relying on the unforgeability of DS. We refer to Lemma 6 for more details. Observe that hybrid $\mathcal{H}_{13}$ corresponds to the original security game, in which the challenger outputs a truly random string during the challenge phase.

Combining the below lemmas, using the triangle inequality, we conclude that for every PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $\left|\Pr[\mathcal{H}_0(\lambda, \mathcal{A}) = 1] - \Pr[\mathcal{H}_{12}(\lambda, \mathcal{A}) = 1]\right| \leq \mathsf{negl}(\lambda)$, where $\mathcal{H}_i(\lambda, \mathcal{A})$ denotes the output of $\mathcal{A}$ in hybrid $\mathcal{H}_i$. □

Lemma 6. *Assuming DS is a strongly unforgeable one-time signature scheme, then for all PPT adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all λ, $|\Pr[\mathcal{H}_0(\lambda, \mathcal{A}) = 1] - \Pr[\mathcal{H}_1(\lambda, \mathcal{A}) = 1]| \leq \mathsf{negl}(\lambda)$.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that is able to distinguish between the hybrids $\mathcal{H}_0$ and $\mathcal{H}_1$. Recall, in the CCA game, the adversary is not allowed decryption queries that are close to the challenge

ciphertext. Therefore, the only way $\mathcal{A}$ can distinguish between the two hybrids is if it makes a decryption query $\mathsf{ct} = \left(c, (c_{i,b})_{i,b}, \sigma\right)$, that is not close to the challenge ciphertext $\mathsf{ct}^* = \left(c^*, \left(c^*_{i,b}\right)_{i,b}, \sigma^*\right)$, where the underlying verification key is equal to $\mathsf{ds.vk}^*$, and the decryption results in a non-$\perp$ output (in particular the signature verification in the last step of decryption passes). We turn $\mathcal{A}$ into an adversary $\mathcal{B}$ that violates the unforgeability of DS. In the first step, the reduction $\mathcal{B}$ obtains a verification key vk^* from the DS challenger. Afterwards, $\mathcal{B}$ proceeds as described in the description of $\mathcal{H}_0$ with the difference that it queries the challenger to generate the signature $\mathsf{ds}.\sigma^*$ for the challenge ciphertext. Next, we specify how the adversary $\mathcal{B}$ will obtain its forgery. Let $\mathsf{ct}' = \left(c', \left(c'_{i,b}\right)_{i,b}, \sigma'\right)$ be the noise-free values recovered at the end of Step 1 in $\mathsf{CheckNoise}$. Let $\mathsf{ds}.x' = \left(c', \left(c'_{i,b}\right)_{i,b}\right)$ (and recall $\mathsf{ds}.x^* = \left(c^*, \left(c^*_{i,b}\right)_{i,b}\right)$ is the string that is signed in the challenge phase). Since $\Delta\left(\mathsf{ct}, \mathsf{ct}^*\right) > \delta\ell_{\mathsf{ct}}$ and $\Delta\left(\mathsf{ct}, \mathsf{ct}'\right) \leq \delta\ell_{\mathsf{ct}}$, [12] we get that $\mathsf{ct}' \neq \mathsf{ct}^*$. We have two cases:

- $\mathsf{ds}.x' \neq \mathsf{ds}.x^*$. In this case, the reduction outputs $\mathsf{ds}.x'$ and $\mathsf{ds}.\sigma$ (output by $\mathsf{CheckNoise}$). This signature verifies (since $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \neq \perp$), and this is a valid forgery since $\mathsf{ds}.x' \neq \mathsf{ds}.x^*$.
- $\mathsf{ds}.x' = \mathsf{ds}.x^*$ but $\sigma' \neq \sigma^*$. We get a strong forgery if we can show that $\mathsf{ds}.\sigma \neq \mathsf{ds}.\sigma^*$. We will show in the claim below that if $\mathsf{ds}.x' = \mathsf{ds}.x^*$ and decryption of ct does not output $\perp$, then the candidate seed t extracted by Find is equal to t^*. From here, we can conclude that $\mathsf{ds}.\sigma \neq \mathsf{ds}.\sigma^*$. [13]

Claim. Let $\mathsf{ct}^* = (\mathsf{ds}.x^*, \sigma^*)$ be the challenge ciphertext (with HPRG seed t^*), and $\mathsf{ct} = (\mathsf{ds}.x, \sigma)$ be any ciphertext such that $\mathsf{Dec}\left(\mathsf{sk}, \mathsf{ct}\right) \neq \perp$. Let $t = \mathsf{Find}(\mathsf{sk}, \mathsf{ct})$, and $\mathsf{ds}.x', \sigma'$ are the error-free terms computed in Step 1 of $\mathsf{CheckNoise}$. If $\mathsf{ds}.x^* = \mathsf{ds}.x'$, then $t = t^*$ (assuming α-adaptive robustness for the base PRC scheme and perfect correctness for the KE scheme and the commitment scheme).

Proof. $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$ does not output $\perp$, therefore $\Delta\left((c_{i,b})_{i,b}, \left(c'_{i,b}\right)_{i,b}\right) \leq \delta\ell_{\mathsf{ct}}$. Consider the set

$$\mathcal{I} = \left\{i : \Delta\left(c_{i,b}, c'_{i,b}\right) \leq \alpha \cdot \ell_{\mathsf{cw}} \text{ for both } b \in \{0,1\}\right\}.$$

Since $\delta = \alpha \cdot \gamma/6$, $|\mathcal{I}| \geq (1-\gamma) \cdot n$.[14] For each of these indices, $\Delta\left(c_{i,b}, c'_{i,b}\right) = \Delta\left(c_{i,b}, c^*_{i,b}\right) \leq \alpha \cdot \ell_{\mathsf{cw}}$. Therefore, by adaptive robustness of the PRC,[15] $\tilde{t}_i = t^*_i$ for all $i \in \mathcal{I}$. Since $\Delta\left(\tilde{t}, t^*\right) \leq \gamma \cdot n$, $t = t^*$. □

□

Lemma 7. *Assuming* Com *satisfies equivocality, then for all PPT adversaries* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all* λ, $|\Pr[\mathcal{H}_1(\lambda, \mathcal{A}) = 1] - \Pr[\mathcal{H}_2(\lambda, \mathcal{A}) = 1]| \leq \mathsf{negl}(\lambda)$.

Proof. The proof follows directly from the equivocality of the commitment scheme. The only distinction between the two hybrids lies in the way $\mathsf{com.pp}$, commitments $(\mathsf{com}^*_i, \mathsf{op}^*_i)_i$ are generated. In $\mathcal{H}_1$, $\mathsf{com.pp}$ is generated using Setup and the commitments and openings are generated using the commitment algorithm, while in $\mathcal{H}_2$, the challenger samples $\left(\mathsf{com.pp}, \left(\mathsf{com}^*_i, \mathsf{op}^*_{i,0}, \mathsf{op}^*_{i,1}\right)_i\right) \leftarrow \mathsf{Equiv}(1^\lambda, 1^n, \mathsf{ds.vk}^*)$ and sets $\mathsf{op}^*_i = \mathsf{op}^*_{i,t^*_i}$. Therefore a PPT adversary that can distinguish between $\mathcal{H}_1$ and $\mathcal{H}_2$ can be used to break the equivocality of Com. □

[12] This is because $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$ does not output $\perp$ in Step 2 of $\mathsf{CheckNoise}$.

[13] If $t = t^*$, then $\sigma^* = \mathsf{PRG}\left(r_{01}\right) \oplus \mathsf{ECC.Encode}(\mathsf{ecc.pp}, \mathsf{ds}.\sigma^*)$ and $\sigma' = \mathsf{PRG}\left(r_{01}\right) \oplus \mathsf{ECC.Encode}(\mathsf{ecc.pp}, \mathsf{ds}.\sigma)$, where $\mathsf{HPRG}(t) = (r_{00}, r_{01}, r_1, \ldots, r_n)$.

[14] Suppose, on the contrary, there are more than $\gamma \cdot n$ indices where at least one out of the two codewords has greater than $\alpha \cdot \ell_{\mathsf{cw}}$ noise. Then the total noise is more than $\gamma \cdot n \cdot (\alpha \cdot \ell_{\mathsf{cw}})$. But this is a contradiction, as we assumed that the total noise is at most $\delta \cdot \ell_{\mathsf{ct}} = (\alpha \cdot \gamma/6)(3 \cdot (2 \cdot n \cdot \ell_{\mathsf{cw}})) = \alpha \cdot \gamma \cdot \ell_{\mathsf{cw}}$. Here, we are using the fact that $|c| = 2 \cdot n \cdot |c_{i,b}| = |\sigma|$.

[15]

Lemma 8. *Assuming perfect correctness of* PKE *and* ECC*, and α-adaptive robustness of* PRC*, then for all PPT adversaries $\mathcal{A}$, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all λ,* $|\Pr[\mathcal{H}_2(\lambda,\mathcal{A})=1]-\Pr[\mathcal{H}_3(\lambda,\mathcal{A})=1]| \leq \mathsf{negl}(\lambda)$.

Proof. Before proving the transition from $\mathcal{H}_2$ to $\mathcal{H}_3$, we prove the following helping lemma.

Lemma 9. *Assuming the commitment scheme satisfies statistical binding, and the PKE scheme satisfies almost perfect-correctness, if* $\mathsf{Dec}(\mathsf{sk},\mathsf{ct}) \neq \perp$*, then for all $i \in [n]$ with $\Delta(c_{i,b}, c'_{i,b}) \leq \alpha\ell_{\mathsf{cw}}$ for both $b \in \{0,1\}$, it holds that $\tilde{t}_i = t_i$.*[16] *Therefore, if* $\mathsf{Dec}(\mathsf{sk},\mathsf{ct}) \neq \perp$*, then $\Delta(t,\tilde{t}) \leq \gamma \cdot n$.*

High level proof sketch. Suppose $\mathsf{Dec}(\mathsf{sk},\mathsf{ct}) \neq \perp$, $\Delta(c_{i,b}, c'_{i,b}) \leq \alpha\ell_{\mathsf{cw}}$ for both $b \in \{0,1\}$, but $\tilde{t}_i = 0$ and $t_i = 1$. Since $\tilde{t}_i$ is set to 0, this means that com_i extracted from $c_{i,0}$ is a commitment of 0, and the corresponding opening is present in $\mathsf{pke.ct}_{i,0}$ (which is also extracted from $c_{i,0}$). However, since $t_i = 1$ and $\mathsf{Dec}(\mathsf{sk},\mathsf{ct}) \neq \perp$, this means that com_i (which is extracted from the header c) is a commitment of 1 (this is checked in $\mathsf{CheckPKE}$). Finally, note that since $\Delta\left(c_{i,b}, c'_{i,b}\right) \leq \alpha\ell_{\mathsf{cw}}$, we also check that the com_i extracted from $c_{i,b}$ is equal to the com_i extracted from the header c. Therefore, if $t_i \neq \tilde{t}_i$, then one must break the statistical binding property of the commitment scheme. The other case ($t_i = 0$, $\tilde{t}_i = 1$) is handled similarly.

Proof. We prove this lemma by contradiction. Suppose there exists an $i \in [n]$ with $\Delta(c_{i,b}, c'_{i,b}) \leq \alpha\ell_{\mathsf{cw}}$ for both $b \in \{0,1\}$ such that $\tilde{t}_i \neq t_i$. We distinguish between two cases: first, $\tilde{t}_i = 0$ and $t_i = 1$ and, second, $\tilde{t}_i = 1$ and $t_i = 0$.
In the first case, since $\tilde{t}_i = 0$, $\mathsf{PRC.Find}$ has checked that $\mathsf{Open}(\mathsf{com.pp}, 0, \underline{\mathsf{com}}_i, \underline{\mathsf{op}}_i, \underline{\mathsf{ds.vk}}_i) = \top$ where $(\underline{\mathsf{pke.ct}}_{i,0}, \underline{\mathsf{com}}_i) = \mathsf{PRC.Decode}(\mathsf{prc.sk}_{i,0}, c_{i,0})$, $\underline{m}_i = \mathsf{PKE.Dec}(\mathsf{pke.sk}_{i,0}, \underline{\mathsf{pke.ct}}_{i,0})$, and $\underline{m}_i = 1\|\underline{\mathsf{op}}_i\|\underline{\mathsf{ds.vk}}_i$. Since we assume that $\Delta(c_{i,b}, c'_{i,b}) \leq \alpha\ell_{\mathsf{cw}}$ for both $b \in \{0,1\}$, during the $\mathsf{PRC.CheckNoise}$ procedure it is checked that $\mathsf{com}_i = \underline{\mathsf{com}}_i$ where $\mathsf{pke.ct}_{i,b}, \mathsf{com}_i$ are obtained from c. Next, we proceed with $\mathsf{PRC.CheckPKE}$, which, since $t_i = 1$, checks that $\mathsf{Open}(\mathsf{com.pp}, 1, \mathsf{com}_i, \mathsf{op}_i, \mathsf{ds.vk}_i) = \top$ where com_i is obtained as part of the output of $\mathsf{PRC.CheckNoise}$. Now, from these checks, we get a contradiction (due to the binding property of the commitment scheme). This concludes the proof of the first case.
Next, we analyze the second case. Since $\tilde{t}_i = 1$, the check in $\mathsf{PRC.Find}$ resulted in $\underline{m}_i \neq 1\|\underline{\mathsf{op}}_i\|\underline{\mathsf{ds.vk}}_i$ or $\mathsf{Open}\left(\mathsf{com.pp}, 0, \underline{\mathsf{com}}_i, \underline{\mathsf{op}}_i, \underline{\mathsf{ds.vk}}_i\right) = \perp$, where $(\underline{\mathsf{pke.ct}}_{i,0}, \underline{\mathsf{com}}_i)$ is obtained from $\mathsf{PRC.Decode}(\mathsf{prc.sk}_{i,0}, c_{i,0})$ and $\underline{m}_i = \mathsf{PKE.Dec}(\mathsf{pke.sk}_{i,0}, \underline{\mathsf{pke.ct}}_{i,0})$. Since we assume that $\Delta(c_{i,b}, c'_{i,b}) \leq \alpha\ell_{\mathsf{cw}}$ for both $b \in \{0,1\}$, during the $\mathsf{PRC.CheckNoise}$ procedure it is checked that $\mathsf{pke.ct}_{i,0} = \underline{\mathsf{pke.ct}}_{i,0}$ and $\mathsf{com}_i = \underline{\mathsf{com}}_i$ where $\mathsf{pke.ct}_{i,b}, \mathsf{com}_i$ is obtained from c. Next, we proceed with $\mathsf{PRC.CheckPKE}$, which, since $t_i = 0$, checks that $m_i = 1\|\mathsf{op}_i\|\mathsf{ds.vk}_i$, $\mathsf{pke.ct}_{i,0} = \mathsf{PKE.Enc}(\mathsf{pke.pk}_{i,0}, m_i; r_i)$ and $\mathsf{Open}\left(\mathsf{com.pp}, 0, \mathsf{com}_i, \mathsf{op}_i, \mathsf{ds.vk}_i\right) = \top$. Assuming the perfect correctness of PKE, we reach a contradiction since the check in Find resulted in $\underline{m}_i \neq 1\|\underline{\mathsf{op}}_i\|\underline{\mathsf{ds.vk}}_i$ or $\mathsf{Open}(\mathsf{com.pp}, 0, \underline{\mathsf{com}}_i, \underline{\mathsf{op}}_i, \underline{\mathsf{ds.vk}}_i) = \perp$ whereas in $\mathsf{CheckPKE}$, we check that $\mathsf{pke.ct}_{i,0}$ is an encryption of $m_i = 1\|\mathsf{op}_i\|\mathsf{ds.vk}_i$ and $\mathsf{Open}(\mathsf{com.pp}, 0, \mathsf{com}_i, \mathsf{op}_i, \mathsf{ds.vk}_i) = \top$. This concludes the proof of the second case.
Having considered the above two cases, it remains to argue the last part of the lemma. From $\mathsf{Dec}(\mathsf{sk},\mathsf{ct}) \neq \perp$, we obtain that $\mathsf{PRC.CheckNoise}$ succeeded without any errors which, in turn, means that $\Delta((c_{i,b}), (c'_{i,b})) \leq \delta\ell_{\mathsf{ct}}$ (follows from Step 2 of $\mathsf{CheckNoise}$). This implies that $\Delta(c_{i,b}, c'_{i,b}) \geq \alpha\ell_{\mathsf{cw}}$ for at most γn indices. Therefore, for at least $(n - \gamma n)$ indices, we have that $\Delta(c_{i,b}, c'_{i,b}) \leq \alpha\ell_{\mathsf{cw}}$ for both $b \in \{0,1\}$. Applying the first half of Lemma 9, we get that $\tilde{t}_i = t_i$ for at least $(n - \gamma n)$ indices. This implies that $\Delta(\tilde{t}, t) \leq n - (n - \gamma n) = \gamma n$. □

The same analysis follow analogously for the $\mathsf{Find}^\dagger$. Therefore, we obtain the following corollary.

Corollary 2. *If* $\mathsf{Dec}^\dagger(\mathsf{sk},\mathsf{ct}) \neq \perp$*, then it must hold that $\Delta(t^\dagger, \tilde{t}^\dagger) \leq \gamma n$ (here, $t^\dagger$ is the output of* $\mathsf{Find}^\dagger$*, and $\tilde{t}^\dagger$ is the output of Step 1 of* $\mathsf{Find}^\dagger$.

Proof. The proof follows as above with the following helping lemma.

Lemma 10. *If* $\mathsf{Dec}^\dagger(\mathsf{sk},\mathsf{ct}) \neq \perp$*, then it follows that for all $i \in [n]$ with $\Delta(c_{i,b}, c'_{i,b}) \leq \alpha\ell_{\mathsf{cw}}$ it holds that $\tilde{t^\dagger}_i = t^\dagger_i$.*

[16] Recall $\tilde{t}$ is produced in Step 1 of Find and t is the output of Find.

□

Next, we proceed with the proof of the transition from $\mathcal{H}_2$ to $\mathcal{H}_3$. The difference between $\mathcal{H}_2$ and $\mathcal{H}_3$ is that the challenger uses Find and $\mathsf{CheckNoise}$ in $\mathcal{H}_2$, and uses $\mathsf{Find}^\dagger$ and $\mathsf{CheckNoise}^\dagger$ along path t^* in $\mathcal{H}_3$. The proof of this Lemma does not rely on the secrecy of t^*. Below, we state a stronger lemma which implies Lemma 8. For any path $\mathfrak{p} \in \{0,1\}^n$, consider a game $\mathcal{G}_\mathfrak{p}$ between a PPT adversary $\mathcal{A}$ and a challenger. The challenger picks a bit $b \in \{0,1\}$. If $b = 0$, the challenger responds to all decryption queries using Find and $\mathsf{CheckNoise}$, else it answers all decryption queries using $\mathsf{Find}^\dagger$ and $\mathsf{CheckNoise}^\dagger$ along path $\mathfrak{p}$. The adversary wins if it can correctly guess b.

Lemma 11. *Assuming perfect correctness of* PKE *and* ECC*, and adaptive robustness of* PRC*, then for any path* $\mathfrak{p} \in \{0,1\}^n$*, for any PPT adversary* $\mathcal{A}$*, there exists a negligible function* negl *such that for all* λ*,*

$$\Pr[\mathcal{A} \text{ wins } \mathcal{G}_\mathfrak{p}(\lambda)] \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

Notations. First we introduce a few notations that will be useful for our proof.

- *differing decryption query* - we call ct a differing decryption query if the decryption of ct using Dec (along path 0^n) is different from decryption of ct using $\mathsf{Find}^\dagger$ and $\mathsf{CheckNoise}^\dagger$(along path $\mathfrak{p}$).
- decryption performed using $\mathsf{Find}^\dagger$ followed by $\mathsf{CheckNoise}^\dagger$ will be referred to as $\mathsf{Dec}^\dagger$. We adopt the convention that all variables defined via $\mathsf{Dec}^\dagger$ will be written with a dagger superscript. For example, $\mathsf{Find}^\dagger$ first computes $\tilde{t^\dagger}$, then decodes it to $s^\dagger$ and outputs $t^\dagger$.

Proof. Suppose there exists a PPT adversary $\mathcal{A}$ that makes Q decryption queries, and can win in game $\mathcal{G}_\mathfrak{p}$ with non-negligible advantage, then it must make a *differing decryption query* ct with non-negligible probability. Let $\mathcal{E}$ denote the event that $\mathcal{A}$ makes a differing decryption query, and $\mathcal{E}_q$ the event that the q^{th} decryption query is the first differing decryption query. Therefore, it suffices to bound $\Pr[\mathcal{E}_q]$ for each q. Fix any $q \in [Q]$, we will show that $\Pr[\mathcal{E}_q] \leq \mathsf{negl}(\lambda)$. We will further split $\mathcal{E}_q$ into the following sub-events:

- $\mathcal{E}_{q0}$: decryption using Find and $\mathsf{CheckNoise}$ outputs $\mathsf{msg} \neq \perp$, $\mathsf{Find}(\mathsf{sk}, \mathsf{ct})$ outputs t, $\mathsf{Find}^\dagger(\mathsf{sk}^\dagger, \mathsf{ct}) = t^\dagger$, and $t \neq t^\dagger$
- $\mathcal{E}_{q1}$: decryption using $\mathsf{Find}^\dagger$ and $\mathsf{CheckNoise}^\dagger$ outputs $\mathsf{msg}^\dagger \neq \perp$, $\mathsf{Find}(\mathsf{sk}, \mathsf{ct})$ outputs t, $\mathsf{Find}^\dagger(\mathsf{sk}^\dagger, \mathsf{ct}) = t^\dagger$, and $t \neq t^\dagger$
- $\mathcal{E}_{q2}$: $\mathsf{Find}(\mathsf{sk}, \mathsf{ct}) = \mathsf{Find}^\dagger\left(\mathsf{sk}^\dagger, \mathsf{ct}\right) = t$, the output of $\mathsf{CheckNoise}$ is not $\perp$, but $\mathsf{CheckNoise}^\dagger$ outputs $\perp$.
- $\mathcal{E}_{q3}$: $\mathsf{Find}(\mathsf{sk}, \mathsf{ct}) = \mathsf{Find}^\dagger\left(\mathsf{sk}^\dagger, \mathsf{ct}\right) = t$, the output of $\mathsf{CheckNoise}$ is $\perp$, but $\mathsf{CheckNoise}^\dagger$ does not output $\perp$.

We can also consider the event $\mathcal{E}_{q4}$ where $\mathsf{Find}(\mathsf{sk}, \mathsf{ct}) = \mathsf{Find}^\dagger\left(\mathsf{sk}^\dagger, \mathsf{ct}\right) = t$ and both $\mathsf{CheckNoise}$ and $\mathsf{CheckNoise}^\dagger$ produce different non-$\perp$ outputs. However, this event never occurs as both $\mathsf{CheckNoise}$ and $\mathsf{CheckNoise}^\dagger$ yield the identical tuple $(\mathsf{msg}, \mathsf{K}, (\mathsf{pke.ct}_{i,b}, \mathsf{com}_i)_{i,b})$ in Step (1), and as both procedures do not abort in Step (3), they necessarily output the same value.
Also, note that if $\mathsf{Find}(\mathsf{sk}, \mathsf{ct}) = \mathsf{Find}^\dagger(\mathsf{sk}^\dagger, \mathsf{ct})$ and $\mathsf{CheckNoise} = \mathsf{CheckNoise}^\dagger$, then the decryption in both scenarios outputs the same value. Therefore, we have $\mathcal{E}_q = \mathcal{E}_{q0} \cup \mathcal{E}_{q1} \cup \mathcal{E}_{q2} \cup \mathcal{E}_{q3}$. Therefore, it suffices to bound $\Pr[\mathcal{E}_{q0}], \Pr[\mathcal{E}_{q1}], \Pr[\mathcal{E}_{q2}]$ and $\Pr[\mathcal{E}_{q3}]$ separately. The events $\mathcal{E}_{q0}$ and $\mathcal{E}_{q1}$ are similar, and likewise $\mathcal{E}_{q2}$ and $\mathcal{E}_{q3}$ are similar. Below we show $\Pr[\mathcal{E}_{q0}]$ and $\Pr[\mathcal{E}_{q2}]$ are negligible.

Bounding the probability of event $\mathcal{E}_{q0}$**.** In this event, decryption using Find and $\mathsf{CheckNoise}$ outputs $\mathsf{msg} \neq \perp$, $\mathsf{Find}(\mathsf{sk}, \mathsf{ct}) = t, \mathsf{Find}^\dagger(\mathsf{sk}^\dagger, \mathsf{ct}) = t^\dagger$, and $t \neq t^\dagger$. We will first show that there exists an index i such that $\tilde{t}_i$ and $\tilde{t_i}^\dagger$ are different,[17] $\mathfrak{p}_i = 1$ and the two PRC codewords $c_{i,0}, c_{i,1}$ have low noise. This is formally captured in the following lemma.

[17]Recall, $\tilde{t}$ (resp. $\tilde{t}^\dagger$) is the noisy versions of t (resp. $t^\dagger$) computed in the first step of Find (resp. $\mathsf{Find}^\dagger$).

Lemma 12. *With overwhelming probability over choice of the public/secret key, if* ct *is any ciphertext such that the following conditions hold:*

- *decryption using* Find *and* $\mathsf{CheckNoise}$ *outputs* $\mathsf{msg} \neq \perp$
- $\mathsf{Find}(\mathsf{sk}, \mathsf{ct}) = t$*, and* $\tilde{t}$ *is the string produced in Step 1 of* Find
- $\mathsf{Find}^\dagger(\mathsf{sk}, \mathsf{ct}) = t^\dagger$ *where* $t^\dagger \neq t$ *(note that* $t^\dagger$ *can be* $\perp$*, but* t *cannot be* $\perp$*). The string* $\widetilde{t^\dagger}$ *is produced in Step 1 of* $\mathsf{Find}^\dagger$

then there exists an index i *such that* $\mathfrak{p}_i = 1$*,* $\tilde{t}_i \neq \widetilde{t^\dagger}_i$*, and for both* $b \in \{0,1\}$*,* $\Delta\left(c_{i,b}, c'_{i,b}\right) \leq \alpha\ell_{\mathsf{cw}}$*. Here,* $c'_{i,b}$ *is the PRC ciphertext computed in Step 1(e) of* $\mathsf{CheckNoise}$*.*

Proof. We will now use the Lemma 9 for proving Lemma 12. Recall, for an overwhelming choice of $(\mathsf{pk}, \mathsf{sk})$, for every $i \in [n]$ and every ct, if $\Delta\left(c_{i,b}, c'_{i,b}\right) \leq \alpha\ell_{\mathsf{cw}}$, then $t_i = \tilde{t}_i$. Fix any such 'good' $(\mathsf{pk}, \mathsf{sk})$. We are given that decryption using Find and $\mathsf{CheckNoise}$ outputs $\mathsf{msg} \neq \perp$, $\mathsf{Find}(\mathsf{sk}, \mathsf{ct}) = t$, $\mathsf{Find}^\dagger(\mathsf{sk}, \mathsf{ct}) = \tilde{t}$ and $t \neq t^\dagger$. Since $\mathsf{CheckNoise}$ does not output $\perp$ in Step 2(a), it follows that $\Delta\left(\left(c, (c_{i,b})_{i,b}, \sigma\right), \left(c', \left(c'_{i,b}\right)_{i,b}, \sigma'\right)\right) \leq \delta\ell_{\mathsf{ct}}$ which, in turn, implies $\Delta\{(c_{i,b})_{i,b}, \left(c'_{i,b}\right)_{i,b}\} \leq \delta\ell_{\mathsf{ct}}$. As a result, if

$$\mathcal{I} = \left\{i : \Delta\left(c_{i,b}, c'_{i,b}\right) \leq \alpha\ell_{\mathsf{cw}} \text{ for both } b \in \{0,1\}\right\}$$

then $|\mathcal{I}| \geq n(1-\gamma)$, and for all $i \in \mathcal{I}$, $t_i = \tilde{t}_i$. We show that there exists an $i \in \mathcal{I}$ such that $\mathfrak{p}_i = 1$ and $\tilde{t}_i \neq \widetilde{t^\dagger}_i$. Assume the contrary, i.e., $\forall i \in \mathcal{I}, \tilde{t}_i = \widetilde{t^\dagger}_i$. (We know that if $\mathfrak{p}_i = 0$, the checks performed in Find and $\mathsf{Find}^\dagger$ for this index are identical.) This implies that $\forall i \in \mathcal{I}, t_i = \widetilde{t^\dagger}_i$. Due to this, we have $\Delta(t, \tilde{t^\dagger}) \leq \gamma n$ and due to perfect correctness of the ECC, $\tilde{t^\dagger}$ will be decoded to t, i.e., $t^\dagger = t$ which is a contradiction to the assumption. □

Using Lemma 12, there exists an i such that $\mathfrak{p}_i = 1$, $\tilde{t}_i \neq \widetilde{t^\dagger_i}$, and for both $b \in \{0,1\}$, $\Delta\left(c_{i,b}, c'_{i,b}\right) \leq \alpha\ell_{\mathsf{cw}}$. We now consider two cases, first, $\tilde{t}_i = 0$ and $\widetilde{t^\dagger_i} = 1$ (we denote this event by $\mathcal{E}_{q0i0}$) and second, $\tilde{t}_i = 1$ and $\widetilde{t^\dagger_i} = 0$ (we denote this event by $\mathcal{E}_{q0i1}$). Note that

$$\mathcal{E}_{q0} = \bigcup_{i \in [n]} \left(\mathcal{E}_{q0i0} \cup \mathcal{E}_{q0i1}\right).$$

We start by analyzing the probability of $\mathcal{E}_{q0i0}$ ($\tilde{t}_i = 0$, $\tilde{t}^\dagger_i = 1$, decryption using Find outputs non-$\perp$, and both $c_{i,0}, c_{i,1}$ have low noise). Let us first make a few observations based on the fact that Find outputs t where $t_i = 0$, $\mathsf{Find}^\dagger$ outputs $t^\dagger$ where $t^\dagger_i = 1$, and $\mathsf{CheckNoise}$ did not output $\perp$.

1. Step 1 of Find sets $\tilde{t}_i = 0$, therefore when $c_{i,0}$ was decoded using $\mathsf{prc.sk}_{i,0}$, it produces $\left(\underline{\mathsf{pke.ct}_{i,0}}, \underline{\mathsf{com}_i}\right)$ where $\underline{\mathsf{pke.ct}_{i,0}}$ decrypts to $\underline{m_i} = 1 \| \underline{\mathsf{op}_i} \| \underline{\mathsf{ds.vk}_i}$ and $\mathsf{Open}\left(\mathsf{com.pp}, 0, \underline{\mathsf{com}_i}, \underline{\mathsf{op}_i}, \underline{\mathsf{ds.vk}_i}\right) = \top$.
2. Likewise, since $\tilde{t}^\dagger_i$ is set to 1 in Step 1 of the decryption using $\mathsf{Find}^\dagger$, and since $\mathfrak{p}_i = 1$, we get that when $c_{i,1}$ is decoded using $\mathsf{prc.sk}_{i,1}$, it produces $\left(\underline{\mathsf{pke.ct}^\dagger_{i,1}}, \underline{\mathsf{com}^\dagger_i}\right)$ where $\underline{\mathsf{pke.ct}^\dagger_{i,1}}$ decrypts to $\underline{m^\dagger_i} = 1 \| \underline{\mathsf{op}^\dagger_i} \| \underline{\mathsf{ds.vk}^\dagger_i}$ and $\mathsf{Open}\left(\mathsf{com.pp}, 1, \underline{\mathsf{com}^\dagger_i}, \underline{\mathsf{op}^\dagger_i}, \underline{\mathsf{ds.vk}^\dagger_i}\right) = \top$.
3. Step 3 of $\mathsf{CheckNoise}$ checks that if $\Delta\left(c_{i,b}, c'_{i,b}\right) \leq \alpha\ell_{\mathsf{cw}}$ for both $b \in \{0,1\}$, then $\mathsf{com}_i = \underline{\mathsf{com}_i}$ and $\mathsf{pke.ct}_{i,0} = \underline{\mathsf{pke.ct}_{i,0}}$, where $\mathsf{pke.ct}_{i,b}$ and com_i are obtained from the payload c.

We will now use the above observations to show that if the above case happens with non-negligible probability, then the base PRC scheme does not satisfy adaptive robustness. First, we observe that with overwhelming

probability over the choice of public parameters, $\underline{\mathsf{com}}_i^\dagger \neq \underline{\mathsf{com}}_i$. This is due to the statistical binding property of the commitment scheme. [18]
Therefore, it only remains to handle the case where $\underline{\mathsf{com}}_i^\dagger \neq \underline{\mathsf{com}}_i$. Note that $\mathsf{com}_i = \underline{\mathsf{com}}_i$, and thus $\underline{\mathsf{com}}_i^\dagger \neq \mathsf{com}_i$. Note that

1. $\Delta(c_{i,1}, c'_{i,1}) \leq \alpha\ell_{\mathsf{cw}}$ (from the description of the first case)
2. $c'_{i,1} = \mathsf{PRC.Encode}\left(\mathsf{prc.pk}_{i,1}, (\mathsf{pke.ct}_{i,1}, \mathsf{com}_i); \mathsf{PRF.Eval}(\mathsf{K}, (i,1))\right)$ (from Step 1e of CheckNoise)
3. $\mathsf{PRC.Decode}\left(\mathsf{prc.sk}_{i,1}, c_{i,1}\right) = \left(\underline{\mathsf{pke.ct}}_{i,1}^\dagger, \underline{\mathsf{com}}_i^\dagger\right) \neq (\mathsf{pke.ct}_{i,1}, \mathsf{com}_i)$.

From here, we get an attack on adaptive robustness. If an adversary $\mathcal{A}$, with non-negligible probability, is able to trigger the event $\mathcal{E}_{q0i0}$, then we can construct an adversary that breaks adaptive robustness of PRC as follows.

1. The PRC challenger sends pk^* to the reduction $\mathcal{B}$.
2. It simulates $\mathcal{H}_2$ by setting $\mathsf{prc.pk}_{i,1} = \mathsf{pk}^*$ and chooses the remaining public/secret keys by itself.
3. On receiving the q-th query $\left(c^{(q)}, \left\{c_{i,b}^{(q)}\right\}, \sigma^{(q)}\right)$ from $\mathcal{A}$, it runs PRC.Find and PRC.CheckNoise to obtain $\left(\mathsf{msg}^{(q)}, \mathsf{K}^{(q)}, \left(\mathsf{pke.ct}_{i,b}^{(q)}, \mathsf{com}_i^{(q)}\right)_{i,b}, \mathsf{ds}.\sigma^{(q)}\right)$ and returns $c_{i,1}^{(q)}$ as the codeword, $\left(\mathsf{pke.ct}_{i,1}^{(q)}, \mathsf{com}_i^{(q)}\right)$ as the message, and $\mathsf{PRF.Eval}(\mathsf{K}^{(q)}, (i,1))$ as the randomness to the challenger.

We observe that if an adversary $\mathcal{A}$ triggers the mentioned event in the q-th query for position i, then the adversary $\mathcal{B}$ forwards a valid adaptive robustness attack to the underlying challenger. Therefore, if the event $\mathcal{E}_{q0i0}$ happens with non-negligible probability, then the reduction also wins the adaptive robustness game with the same probability.

Next, we analyze the event $\mathcal{E}_{q0i1}$ ($\tilde{t}_i = 1$, $\tilde{t}_i^\dagger = 0$, decryption using Find outputs non-$\perp$, and both $c_{i,0}, c_{i,1}$ have low noise). Again, we first make a few observations based on the fact that Find outputs t where $t_i = 1$, $\mathsf{Find}^\dagger$ outputs $t^\dagger$ where $t_i^\dagger = 0$, and CheckNoise did not output $\perp$.

1. Step 1 of Find sets $\tilde{t}_i = 1$, therefore when $c_{i,0}$ is decoded using $\mathsf{prc.sk}_{i,0}$, it produces $\underline{\mathsf{pke.ct}}_{i,0}$ and $\underline{\mathsf{com}}_i$, and when $\underline{\mathsf{pke.ct}}_{i,0}$ is decrypted using $\mathsf{pke.sk}_{i,0}$, either it does not produce a decryption of the form $\underline{m}_i = 1\|\underline{\mathsf{op}}_i\|\underline{\mathsf{ds.vk}}_i$, or $\mathsf{Open}\left(\mathsf{com.pp}, 0, \underline{\mathsf{com}}_i, \underline{\mathsf{op}}_i, \underline{\mathsf{ds.vk}}_i\right) = \perp$.
2. Step 3 of CheckNoise checks that if $\Delta(c_{i,b}, c'_{i,b}) \leq \alpha\ell_{\mathsf{cw}}$ for both $b \in \{0,1\}$, then $\mathsf{com}_i = \underline{\mathsf{com}}_i$ and $\mathsf{pke.ct}_{i,1} = \underline{\mathsf{pke.ct}}_{i,1}$, where $\mathsf{pke.ct}_{i,b}$ and com_i are obtained from the payload c.
3. In Step 2(b) of CheckPKE, it is checked that $m_i = 1\|\mathsf{op}_i\|\mathsf{ds.vk}_i$ and $\mathsf{Open}\left(\mathsf{com.pp}, 1, \mathsf{com}_i, \mathsf{op}_i, \mathsf{ds.vk}_i\right) = \top$ where $m_i = \mathsf{PKE.Recover}(\mathsf{pke.pk}_{i,1}, \mathsf{pke.ct}_{i,1}, r_i)$ and also checked that $\mathsf{pke.ct}_{i,1} = \mathsf{PKE.Enc}(\mathsf{pke.pk}_{i,1}, m_i; r_i)$.
4. Next, $\mathsf{Find}^\dagger$ sets $\tilde{t}_i = 0$ and $\mathfrak{p}_i = 1$, either $\underline{m}_i^\dagger$ is not of the form $1\|\underline{\mathsf{op}}_i^\dagger\|\underline{\mathsf{ds.vk}}_i^\dagger$ or $\mathsf{Open}\left(\mathsf{com.pp}, 1, \underline{\mathsf{com}}_i^\dagger, \underline{\mathsf{op}}_i^\dagger, \underline{\mathsf{ds.vk}}_i^\dagger\right) = \perp$ (here $\underline{\mathsf{pke.ct}}_{i,1}^\dagger, \underline{\mathsf{com}}_i^\dagger$ is obtained from $\mathsf{PRC.Decode}(\mathsf{prc.sk}_{i,1}, c_{i,1})$ and $\underline{m}_i^\dagger$ from $\mathsf{PKE.Dec}\left(\mathsf{pke.sk}_{i,1}, \underline{\mathsf{pke.ct}}_{i,1}^\dagger\right)$).
5. If the pair $(\underline{\mathsf{pke.ct}}_{i,1}^\dagger, \underline{\mathsf{com}}_i^\dagger)$ (which is obtained by decrypting $c_{i,1}$ using $\mathsf{prc.sk}_{i,1}$) is equal to the pair $(\mathsf{pke.ct}_{i,1}, \mathsf{com}_i)$ (obtained from the payload c in the CheckNoise procedure) then this violates the perfect correctness. This follows from the observations listed above.

Therefore, it only remains to handle the case where $(\underline{\mathsf{pke.ct}}_{i,1}^\dagger, \underline{\mathsf{com}}_i^\dagger) \neq (\mathsf{pke.ct}_{i,1}, \mathsf{com}_i)$. Since, $\Delta(c_{i,1}, c'_{i,1}) \leq \alpha\ell_{\mathsf{cw}}$ and the fact that $c'_{i,1}$ is an encoding of $(\mathsf{pke.ct}_{i,1}, \mathsf{com}_i)$ computed in Step 1 of CheckNoise, then we use adaptive robustness to show that this event happens with negligible probability. More precisely, if an adversary $\mathcal{A}$, with non-negligible probability, is able to trigger this event, then we can construct an adversary that breaks adaptive robustness of PRC as follows.

[18] Note that the commitments can potentially be for different tags. However, since the commitment to 0 is independent of the tag, the statistical binding property is violated.

1. The PRC challenger outputs pk^* to the reduction $\mathcal{B}$.
2. It simulates $\mathcal{H}_2$ by setting $\mathsf{prc.pk}_{i,1} = \mathsf{pk}^*$ (and samples the remaining public/secret keys by itself).
3. On receiving the q-th query $\left(c^{(q)}, \left\{c_{i,b}^{(q)}\right\}, \sigma^{(q)}\right)$ from $\mathcal{A}$, it runs $\mathsf{PRC.Find}$ and $\mathsf{PRC.CheckNoise}$ to obtain $\left(\mathsf{msg}^{(q)}, \mathsf{K}^{(q)}, \left(\mathsf{pke.ct}_{i,b}^{(q)}, \mathsf{com}_i^{(q)}\right)_{i,b}, \mathsf{ds}.\sigma^{(q)}\right)$ and returns $c_{i,1}^{(q)}$ as the codeword, $\left(\mathsf{pke.ct}_{i,1}^{(q)}, \mathsf{com}_i^{(q)}\right)$ as the message, and $\mathsf{PRF.Eval}(\mathsf{K}^{(q)}, (i,1))$ as the randomness to the challenger.

We observe that if an adversary $\mathcal{A}$ triggers the mentioned event in the q-th query for position i, then the adversary $\mathcal{B}$ forwards a valid adaptive robustness attack to the underlying challenger. Given that this only happens if $\mathcal{B}$ guesses the correct index i and query q, this introduces an additional security loss of $n \cdot Q$.

Bounding the probability of event $\mathcal{E}_{q1}$. In this event, decryption using $\mathsf{Find}^\dagger$ and $\mathsf{CheckNoise}^\dagger$ outputs $\mathsf{msg}^\dagger \neq \perp$, $\mathsf{Find}(\mathsf{sk}, \mathsf{ct}) = t$, and $t \neq t^\dagger$. This case is analogous to $\mathcal{E}_{q0}$ and the proof is almost identical; the lemmas and claims will be with respect to decryption using $\mathsf{Find}^\dagger$ and $\mathsf{CheckNoise}^\dagger$.

Bounding the probability of event $\mathcal{E}_{q2}$. In this event, both Find and $\mathsf{Find}^\dagger$ output the same string t, $\mathsf{CheckNoise}$ does not output $\perp$, but $\mathsf{CheckNoise}^\dagger$ outputs $\perp$. Since Find and $\mathsf{Find}^\dagger$ output the same t, the computation in Step 1 of $\mathsf{CheckNoise}$ and $\mathsf{CheckNoise}^\dagger$ is identical, and therefore,

- $(\mathsf{com}_i)_i = \left(\mathsf{com}_i^\dagger\right)_i$,
- $(\mathsf{pke.ct}_{i,b})_{i,b} = \left(\mathsf{pke.ct}_{i,b}^\dagger\right)_{i,b}$,
- $\left(c'_{i,b}\right)_{i,b} = \left(c_{i,b}^{\dagger'}\right)_{i,b}$

However, the difference in $\mathsf{CheckNoise}$ and $\mathsf{CheckNoise}^\dagger$ can arise from Step 3. $\mathsf{CheckNoise}$ decrypts $(c_{i,0})_i$ using the secret keys $(\mathsf{prc.sk}_{i,0})_i$ and obtains $\left(\underline{\mathsf{pke.ct}_{i,0}}, \underline{\mathsf{com}_i}\right)_i$ while $\mathsf{CheckNoise}^\dagger$ decrypts $(c_{i,\mathsf{p}_i})_i$ using the secret keys $(\mathsf{prc.sk}_{i,\mathsf{p}_i})_i$ and obtains $\left(\underline{\mathsf{pke.ct}}^\dagger{}_{i,\mathsf{p}_i}, \underline{\mathsf{com}_i^\dagger}\right)_i$. Since $\mathsf{CheckNoise}$ does not output $\perp$ but $\mathsf{CheckNoise}^\dagger$ outputs $\perp$, there exists an index i such that

- $\mathsf{p}_i = 1$
- $\Delta\left(c_{i,b}, c'_{i,b}\right) \leq \delta\ell_{\mathsf{ct}}$ for both $b \in \{0,1\}$
- $\Delta\left(c_{i,b}, c_{i,b}^{\dagger'}\right) \leq \delta\ell_{\mathsf{ct}}$ for both $b \in \{0,1\}$ (since $c'_{i,b} = c_{i,b}^{\dagger'}$)
- $\mathsf{PRC.Decode}\left(\mathsf{prc.sk}_{i,0}, c_{i,0}\right) = \left(\underline{\mathsf{pke.ct}_{i,0}}, \underline{\mathsf{com}_i}\right) = (\mathsf{pke.ct}_{i,0}, \mathsf{com}_i)$ (since $\mathsf{CheckNoise}$ does not output $\perp$ in Step 3b)
- $\mathsf{PRC.Decode}\left(\mathsf{prc.sk}_{i,\mathsf{p}_i}, c_{i,\mathsf{p}_i}\right) = \left(\underline{\mathsf{pke.ct}}^\dagger{}_{i,\mathsf{p}_i}, \underline{\mathsf{com}_i^\dagger}\right) \neq (\mathsf{pke.ct}_{i,\mathsf{p}_i}, \mathsf{com}_i)$ (since $\mathsf{CheckNoise}^\dagger$ outputs $\perp$ in Step 3b)

Using the first, third and fifth lines above, we get that

- $\Delta\left(c_{i,1}, c_{i,1}^{\dagger'}\right) \leq \alpha\ell_{\mathsf{cw}}$,
- $\mathsf{PRC.Decode}\left(\mathsf{prc.sk}_{i,1}, c_{i,1}\right) \neq \mathsf{PRC.Decode}\left(\mathsf{prc.sk}_{i,1}, c_{i,1}^{\dagger'}\right)$.

Therefore, we obtain an attack on adaptive robustness if event $\mathcal{E}_{q2}$ happens with non-negligible probability. The reduction algorithm receives the PRC public key $\mathsf{prc.pk}^*$ from the challenger. It samples $2n-1$ PRC public/secret keys, and sets $\mathsf{prc.pk}_{i,\mathsf{p}_i} = \mathsf{prc.pk}^*$. For the first $q-1$ decryption queries, it uses $(\mathsf{prc.sk}_{i,0}, \mathsf{pke.sk}_{i,0})$ to decrypt (note that the first $q-1$ decryptions can be performed along either path p or the path 0^n). On receiving the q^{th} decryption query $(c, (c_{i,0}, c_{i,1}))$, the reduction first computes $t = \mathsf{Find}\left((\mathsf{sk}_{i,0})_i, (c_{i,0})_i\right)$. Next, it computes the various components in Step 1 of $\mathsf{CheckNoise}$ (which are same as the components computed

in Step 1 of $\mathsf{CheckNoise}^{\dagger}$). The reduction finds an i that satisfies the five conditions listed above. It sends $c_{i,1}$ as the noisy codeword, $(\mathsf{pke.ct}_{i,\mathsf{p}_i}, \mathsf{com}_i)$ as the message and $\mathsf{PRF.Eval}(\mathsf{K}, (i,1))$ as the randomness to the adaptive robustness challenger.

□
□

Lemma 13. *Assuming* PKE *is IND-CPA secure, then for all PPT adversaries $\mathcal{A}$, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all* λ, $|\Pr[\mathcal{H}_3(\lambda,\mathcal{A})=1] - \Pr[\mathcal{H}_4(\lambda,\mathcal{A})=1]| \leq \mathsf{negl}(\lambda)$.

Proof. To prove the transition from $\mathcal{H}_3$ to $\mathcal{H}_4$, we introduce a set of intermediate hybrids $\mathcal{H}_{3,0},\ldots,\mathcal{H}_{3,n}$. Here, $\mathcal{H}_{3,j}$ is defined as the hybrid where the first j ciphertexts $\mathsf{pke.ct}^*_{1,\overline{t^*_1}},\ldots,\mathsf{pke.ct}^*_{j,\overline{t^*_j}}$ are generated as $\mathsf{pke.ct}^*_{i,\overline{t^*_i}} := \mathsf{PKE.Enc}(\mathsf{pke.pk}_{i,\overline{t^*_i}}, 1\|\mathsf{op}^*_{i,\overline{t^*_i}}\|\mathsf{ds.vk}^*; r^*_i)$ for all $i\in[j]$ and the remaining ciphertexts $\mathsf{pke.ct}^*_{j+1,\overline{t^*_{j+1}}},\ldots,\mathsf{pke.ct}^*_{n,\overline{t^*_n}}$ are generated as $\mathsf{pke.ct}^*_{i,\overline{t^*_i}} := \mathsf{PKE.Enc}(\mathsf{pke.pk}_{i,\overline{t^*_i}}, 0^{1+\ell_{\mathsf{vk}}+\ell_{\mathrm{com}}}; r^*_i)$ for all $i\in[n]\setminus[j]$. To conclude the transition, it remains to show that $\mathcal{H}_{3,i-1} \approx_c \mathcal{H}_{3,i}$ for all $i\in[n]$, which we prove for a general index $i^\star$ in the next step.
We prove the transition from $\mathcal{H}_{3,i^\star-1}$ to $\mathcal{H}_{3,i^\star}$ by contradiction. In more detail, we assume that there exists an adversary $\mathcal{A}$ that is able to distinguish between the hybrids $\mathcal{H}_{3,i^\star-1}$ and $\mathcal{H}_{3,i^\star}$ and turn it into an adversary $\mathcal{B}$ that violates the IND-CPA security of the PKE scheme. The reduction $\mathcal{B}$ proceeds as follows, it proceeds as described in the initialization of $\mathcal{H}_3$ until the generation of the ciphertexts, with the exception that it does not generate the public key $\mathsf{pke.pk}_{i^\star,\overline{t^*_{i^\star}}}$ itself but receives it from the PKE challenger. Now, we distinguish between three cases: first, the ciphertexts $\{\mathsf{pke.ct}_{i,\overline{t^*_i}}\}_{i\in[i^\star-1]}$, second, $\mathsf{pke.ct}_{i^\star,\overline{t_{i^\star}}}$, and, third, $\{\mathsf{pke.ct}_{i,\overline{t^*_i}}\}_{i\in[n]\setminus[i^\star]}$. The ciphertexts $\{\mathsf{pke.ct}_{i,\overline{t^*_i}}\}_{i\in[i^\star-1]}$ are generated as $\{\mathsf{PKE.Enc}(\mathsf{pke.pk}_{i,\overline{t^*_i}}, 1\|\mathsf{op}^*_{i,\overline{t^*_i}}\|\mathsf{ds.vk}^*; r^*_i)\}_{i\in[i^\star-1]}$ by $\mathcal{B}$ and the ciphertexts $\{\mathsf{pke.ct}_{i,\overline{t^*_i}}\}_{i\in[n]\setminus[i^\star]}$ are generated as $\{\mathsf{PKE.Enc}(\mathsf{pke.pk}_{i,\overline{t^*_i}}, 0^{1+\ell_{\mathsf{vk}}+\ell_{\mathrm{com}}}; r^*_i)\}_{i\in[n]\setminus[i^\star]}$ by $\mathcal{B}$. To generate the ciphertext $\mathsf{pke.ct}_{i^\star,\overline{t^*_{i^\star}}}$, $\mathcal{B}$ submits $((0^{1+\ell_{\mathsf{vk}}+\ell_{\mathrm{com}}}), (1\|\mathsf{op}^*_{i,\overline{t^*_i}}\|\mathsf{ds.vk}^*))$ to its underlying challenger and receives $\mathsf{pke.ct}_{i^\star,\overline{t^*_{i^\star}}}$ in reply. Afterwards, $\mathcal{B}$ sends all those ciphertexts to the adversary $\mathcal{A}$ and proceeds as described in $\mathcal{H}_3$.
We observe that if the underlying challenger encrypts $0^{1+\ell_{\mathsf{vk}}+\ell_{\mathrm{com}}}$ as the ciphertext $\mathsf{pke.ct}_{i^\star,\overline{t^*_{i^\star}}}$, then $\mathcal{B}$ simulates hybrid $\mathcal{H}_{3,i^\star-1}$ towards $\mathcal{A}$. In the case that the underlying challenger encrypts $1\|\mathsf{op}^*_{i,\overline{t^*_i}}\|\mathsf{ds.vk}^*$ as the ciphertext $\mathsf{pke.ct}_{i^\star,\overline{t^*_{i^\star}}}$, then $\mathcal{B}$ simulates hybrid $\mathcal{H}_{3,i^\star}$ towards $\mathcal{A}$. This results in a perfect reduction and, therefore, the success probability of $\mathcal{A}$ is being bounded by the security advantage of IND-CPA of PKE. Since the above reduction works for all $i^\star\in[n]$, we obtain the indistingushability between the hybrids $\mathcal{H}_3$ and $\mathcal{H}_4$, bounded by n times the security advantage of IND-CPA. □

Lemma 14. *Assuming the security of the hinting PRG and the PRG security, then for all PPT adversaries $\mathcal{A}$, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all* λ, $|\Pr[\mathcal{H}_5(\lambda,\mathcal{A})=1] - \Pr[\mathcal{H}_6(\lambda,\mathcal{A})=1]| \leq \mathsf{negl}(\lambda)$.

Proof. To prove this lemma, we introduce an intermediate hybrid $\mathcal{H}_{5.5}$. This hybrid is identical to $\mathcal{H}_5$, except that the values $\tilde{r}^*_i$ are sampled uniformly at random instead of being generated by the hinting PRG.
We will first show that $\mathcal{H}_5$ and $\mathcal{H}_{5.5}$ are indistinguishable by contradiction. In more detail, we assume that there exists an adversary $\mathcal{A}$ that is able to distinguish between the hybrids $\mathcal{H}_5$ and $\mathcal{H}_{5.5}$ and turn it into an adversary $\mathcal{B}$ that violates the security of the hinting PRG. The reduction $\mathcal{B}$ proceeds as follows, in the first step, it receives $\mathsf{hprg.pp}$ and $\tilde{r}^*_0, \{\tilde{r}^*_{i,b}\}_{i\in[n],b\in\{0,1\}}$ from the challenger. It then proceeds as described in the description of $\mathcal{H}_5$ until the adversary $\mathcal{A}$ asks its challenge encryption query using a message m. To generate the corresponding PKE ciphertexts, the adversary $\mathcal{B}$ proceeds as described in $\mathcal{H}_5$ while it uses the values $\tilde{r}^*_0, \{\tilde{r}^*_{i,b}\}_{i\in[n],b\in\{0,1\}}$ obtained from the challenger instead of sampling them itself. Afterwards, $\mathcal{B}$ proceeds again as in the description of $\mathcal{H}_5$.
We observe that if the underlying challenger generates the values $\tilde{r}^*_0$ and $\{\tilde{r}^*_{i,b}\}_{i\in[n],b\in\{0,1\}}$ using the hinting PRG HPRG, then $\mathcal{B}$ simulates hybrid $\mathcal{H}_5$ towards $\mathcal{A}$. In the case that the underlying challenger samples the values $\tilde{r}^*_0$ and $\{\tilde{r}^*_{i,b}\}_{i\in[n],b\in\{0,1\}}$ at random, $\mathcal{B}$ simulates $\mathcal{H}_{5.5}$ towards $\mathcal{A}$. This results in a perfect reduction and, therefore, the success probability of $\mathcal{A}$ is being bounded by the security advantage of HPRG.

To show $\mathcal{H}_6$ and $\mathcal{H}_{5.5}$ are indistinguishable, we rely on the security of the PRG. The only distinction lies in the way c^* and σ^* are generated. In $\mathcal{H}_{5.5}$, c^* is set to $\mathsf{PRG}(\tilde{r_0^*}[0]) + \mathsf{ECC}(m^*, \mathsf{K}^*, \{\mathsf{pke.ct}^*_{i,b}, c^*_{i,2}\}_{i\in[n],b\in\{0,1\}}) \cdot B_{\mathsf{pl}}$ and σ^* is set to $\mathsf{PRG}(\tilde{r_0^*}[1]) \oplus \mathsf{ECC.Encode}(\mathsf{ecc.pp}, \mathsf{ds}.\sigma^*)$. Whereas, in $\mathcal{H}_6$, they are sampled uniformly at random. Since, the value $\tilde{r}_0^*$ is not used elsewhere in either experiment, the security of PRG guarantees that no PPT adversary can distinguish between $\mathcal{H}_{5.5}$ and $\mathcal{H}_6$ with more than negligible advantage. □

Lemma 15. *Assuming the security of the PRF, then for all PPT adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all λ, $|\Pr[\mathcal{H}_6(\lambda,\mathcal{A})=1] - \Pr[\mathcal{H}_7(\lambda,\mathcal{A})=1]| \leq \mathsf{negl}(\lambda)$.*

Proof. We prove this lemma by contradiction. In more detail, we assume that there exists an adversary $\mathcal{A}$ that is able to distinguish between the hybrids $\mathcal{H}_6$ and $\mathcal{H}_7$ and turn it into an adversary $\mathcal{B}$ that violates the security of PRF. The reduction $\mathcal{B}$ proceeds as follows - in the initialization step, it proceeds as described in the description of $\mathcal{H}_6$ until the generation of the PRC ciphertexts. The adversary $\mathcal{B}$ sends (i,b) for all $i \in [n], b \in \{0,1\}$ and obtains from the PRF challenger, $u_{i,b}$ instead of sampling them itself to generate $c^\star_{i,b}$. Afterwards, $\mathcal{B}$ proceeds again as in the description of $\mathcal{H}_6$. It is important to note that the PRF key is not required because c^* is sampled uniformly at random.
We observe that if the underlying challenger generates the values $u_{i,b}$ using the PRF, then $\mathcal{B}$ simulates hybrid $\mathcal{H}_6$ towards $\mathcal{A}$. In the case that the underlying challenger samples the values at random, $\mathcal{B}$ simulates $\mathcal{H}_7$ towards $\mathcal{A}$. This results in a perfect reduction and, therefore, the success probability of $\mathcal{A}$ is being bounded by the security advantage of PRF. □

Lemma 16. *Assuming PRC is IND-CPA secure, then for all PPT adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all λ, $|\Pr[\mathcal{H}_7(\lambda,\mathcal{A})=1] - \Pr[\mathcal{H}_8(\lambda,\mathcal{A})=1]| \leq \mathsf{negl}(\lambda)$.*

Proof. To prove the transition from $\mathcal{H}_7$ to $\mathcal{H}_8$, we introduce a set of intermediate hybrids $\mathcal{H}_{7,0}, \ldots, \mathcal{H}_{7,n}$. Here, $\mathcal{H}_{7,j}$ is defined as the hybrid where the first j ciphertexts $c^*_{1,1}, \ldots, c^*_{j,1}$ are generated as random values, i.e., $c^*_{i,1} \leftarrow \{0,1\}^{\ell_{\mathsf{ct}}}$ for all $i \in [j]$, and the remaining ciphertexts $c^*_{j+1,1}, \ldots, c^*_{n,1}$ are generated as $c^*_{i,1} \coloneqq \mathsf{PRC.Encode}(\mathsf{prc.pk}_{i,1}, (\mathsf{pke.ct}_{i,1}, \mathsf{com}_i)); u_{i,1})$ for all $i \in [n] \setminus [j]$. To conclude the transition, it remains to show that $\mathcal{H}_{7,i-1} \approx_c \mathcal{H}_{7,i}$ for all $i \in [n]$, which we prove for an index $i^\star$ in the next step.
We prove the transition from $\mathcal{H}_{7,i^\star-1}$ to $\mathcal{H}_{7,i^\star}$ by contradiction. In more detail, we assume that there exists an adversary $\mathcal{A}$ that is able to distinguish between the hybrids $\mathcal{H}_{7,i^\star-1}$ and $\mathcal{H}_{7,i^\star}$ and turn it into an adversary $\mathcal{B}$ that violates the $\mathsf{IND\text{-}CPA}$ security of the PRC scheme. The reduction $\mathcal{B}$ proceeds as follows, it proceeds as described in the initialization of $\mathcal{H}_7$ until the generation of the ciphertexts, with the exception that it does not generate the public key $\mathsf{prc.pk}_{i^\star,1}$ itself but receives it from the underlying challenger. Now, we distinguish between three cases: first, the ciphertexts $\{c^\star_{i,1}\}_{i\in[i^\star-1]}$, second, $c^\star_{i^\star,1}$, and, third, $\{c^\star_{i,1}\}_{i\in[n]\setminus[i^\star-1]}$. The ciphertexts $\{c^\star_{i,1}\}_{i\in[i^\star-1]}$ are sampled at random, i.e., $c^*_{i,1} \leftarrow \{0,1\}^{\ell_{\mathsf{ct}}}$ for all $i \in [i^\star - 1]$, by $\mathcal{B}$ and the ciphertexts $\{c^\star_{i,1}\}_{i\in[n]\setminus[i^\star]}$ are generated as $\{\mathsf{PRC.Encode}(\mathsf{prc.pk}_{i,1}, (\mathsf{pke.ct}^\star_{i,1}, \mathsf{com}^\star_i); u_{i,1})\}_{i\in[n]\setminus[i^\star]}$ by $\mathcal{B}$. To generate the ciphertext $c^\star_{i^\star,1}$, $\mathcal{B}$ submits $(\mathsf{pke.ct}_{i^\star,1}, \mathsf{com}_{i^\star})$ to the PRC challenger and receives $c_{i^\star,1}$ in reply. Afterwards, $\mathcal{B}$ sends all those ciphertexts to the adversary $\mathcal{A}$ and proceeds as described in $\mathcal{H}_7$.
We observe that if the underlying challenger encodes $(\mathsf{pke.ct}_{i^\star,1}, \mathsf{com}_{i^\star})$ as the ciphertext $c^\star_{i^\star,1}$, then $\mathcal{B}$ simulates hybrid $\mathcal{H}_{7,i^\star-1}$ towards $\mathcal{A}$. In the case that the underlying challenger sends a random value $c^\star_{i^\star,1} \leftarrow \{0,1\}^{\ell_{\mathsf{ct}}}$ as the ciphertext, then $\mathcal{B}$ simulates hybrid $\mathcal{H}_{7,i^\star}$ towards $\mathcal{A}$. This results in a perfect reduction and, therefore, the success probability of $\mathcal{A}$ is being bounded by the security advantage of $\mathsf{IND\text{-}CPA}$ of PRC. Since the above reduction works for all $i^\star \in [n]$, we obtain the indistinguishability between the hybrids $\mathcal{H}_7$ and $\mathcal{H}_8$, bounded by n times the security advantage of $\mathsf{IND\text{-}CPA}$. □

Acknowledgements

We thank the anonymous Crypto 2026 reviewers for their helpful feedback, and in particular one reviewer for suggesting Raw Reed–Solomon codes as a useful cryptanalytic target for noisy McEliece-type assumptions.

Nico Döttling and Mahesh Sreekumar Rajasree: Funded by the European Union (ERC, LACONIC, 101041207). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them. Neither the European Union nor the granting authority can be held responsible for them. This work has been partly done while the third author was at IIT Delhi Abu Dhabi and the CISPA Helmholtz Center for Information Security.

References

AAC+25. Omar Alrabiah, Prabhanjan Ananth, Miranda Christ, Yevgeniy Dodis, and Sam Gunn. Ideal pseudorandom codes. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, STOC '25, page 1638–1647, New York, NY, USA, 2025. Association for Computing Machinery.

ACPS09. Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In Shai Halevi, editor, *Advances in Cryptology – CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 595–618, Santa Barbara, CA, USA, August 16–20, 2009. Springer Berlin Heidelberg, Germany.

Ale03. Michael Alekhnovich. More on average case vs approximation complexity. In *44th Annual Symposium on Foundations of Computer Science*, pages 298–307, Cambridge, MA, USA, October 11–14, 2003. IEEE Computer Society Press.

AP22. Navid Alamati and Sikhar Patranabis. Cryptographic primitives with hinting property. In Shweta Agrawal and Dongdai Lin, editors, *Advances in Cryptology – ASIACRYPT 2022, Part I*, volume 13791 of *Lecture Notes in Computer Science*, pages 33–62, Taipei, Taiwan, December 5–9, 2022. Springer, Cham, Switzerland.

BBDP01. Mihir Bellare, Alexandra Boldyreva, Anand Desai, and David Pointcheval. Key-privacy in public-key encryption. In Colin Boyd, editor, *Advances in Cryptology – ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 566–582, Gold Coast, Australia, December 9–13, 2001. Springer Berlin Heidelberg, Germany.

BBP04. Mihir Bellare, Alexandra Boldyreva, and Adriana Palacio. An uninstantiable random-oracle-model scheme for a hybrid-encryption problem. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 171–188, Interlaken, Switzerland, May 2–6, 2004. Springer Berlin Heidelberg, Germany.

BC24. Daniel J. Bernstein and Tung Chou. CryptAttackTester: high-assurance attack analysis. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology – CRYPTO 2024, Part VI*, volume 14925 of *Lecture Notes in Computer Science*, pages 141–182, Santa Barbara, CA, USA, August 18–22, 2024. Springer, Cham, Switzerland.

BCHK07. Dan Boneh, Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. *SIAM J. Comput.*, 36(5):1301–1328, 2007.

BDD22. Pedro Branco, Nico Döttling, and Jesko Dujmovic. Rate-1 incompressible encryption from standard assumptions. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022: 20th Theory of Cryptography Conference, Part II*, volume 13748 of *Lecture Notes in Computer Science*, pages 33–69, Chicago, IL, USA, November 7–10, 2022. Springer, Cham, Switzerland.

BG10. Zvika Brakerski and Shafi Goldwasser. Circular and leakage resilient public-key encryption under subgroup indistinguishability - (or: Quadratic residuosity strikes back). In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 1–20, Santa Barbara, CA, USA, August 15–19, 2010. Springer Berlin Heidelberg, Germany.

BHHO08. Dan Boneh, Shai Halevi, Michael Hamburg, and Rafail Ostrovsky. Circular-secure encryption from decision Diffie-Hellman. In David Wagner, editor, *Advances in Cryptology – CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 108–125, Santa Barbara, CA, USA, August 17–21, 2008. Springer Berlin Heidelberg, Germany.

BK05. Dan Boneh and Jonathan Katz. Improved efficiency for CCA-secure cryptosystems built using identity-based encryption. In Alfred Menezes, editor, *Topics in Cryptology – CT-RSA 2005*, volume 3376 of *Lecture Notes in Computer Science*, pages 87–103, San Francisco, CA, USA, February 14–18, 2005. Springer Berlin Heidelberg, Germany.

BKR23. Andrej Bogdanov, Pravesh K Kothari, and Alon Rosen. Public-key encryption, local pseudorandom generators, and the low-degree method. In *Theory of Cryptography Conference*, pages 268–285. Springer, 2023.

BLP08. Daniel J Bernstein, Tanja Lange, and Christiane Peters. Attacking and defending the McEliece cryptosystem. In *International Workshop on Post-Quantum Cryptography*, pages 31–46. Springer, 2008.

BLSV18. Zvika Brakerski, Alex Lombardi, Gil Segev, and Vinod Vaikuntanathan. Anonymous IBE, leakage resilience and circular security from new assumptions. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018, Part I*, volume 10820 of *Lecture Notes in Computer Science*, pages 535–564, Tel Aviv, Israel, April 29 – May 3, 2018. Springer, Cham, Switzerland.

BR93. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby, editors, *ACM CCS 93: 1st Conference on Computer and Communications Security*, pages 62–73, Fairfax, Virginia, USA, November 3–5, 1993. ACM Press.

CC98. Anne Canteaut and Florent Chabaud. A new algorithm for finding minimum-weight words in a linear code: Application to McEliece's cryptosystem and to narrow-sense BCH codes of length 511. *IEEE Transactions on Information Theory*, 44(1):367–378, 1998.

CG24. Miranda Christ and Sam Gunn. Pseudorandom error-correcting codes. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology – CRYPTO 2024, Part VI*, volume 14925 of *Lecture Notes in Computer Science*, pages 325–347, Santa Barbara, CA, USA, August 18–22, 2024. Springer, Cham, Switzerland.

CGG+25. Miranda Christ, Noah Golowich, Sam Gunn, Ankur Moitra, and Daniel Wichs. Improved Pseudorandom Codes from Permuted Puzzles. Cryptology ePrint Archive, Paper 2025/2222, 2025. To appear in STOC 2026.

CGGU+14. Alain Couvreur, Philippe Gaborit, Valérie Gauthier-Umaña, Ayoub Otmani, and Jean-Pierre Tillich. Distinguisher-based attacks on public-key cryptosystems using Reed–Solomon codes. *Designs, Codes and Cryptography*, 73(2):641–666, 2014.

CHK03. Ran Canetti, Shai Halevi, and Jonathan Katz. A forward-secure public-key encryption scheme. In Eli Biham, editor, *Advances in Cryptology – EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 255–271, Warsaw, Poland, May 4–8, 2003. Springer Berlin Heidelberg, Germany.

CHS25. Aloni Cohen, Alexander Hoover, and Gabe Schoenbach. Watermarking language models for many adaptive users. In Marina Blanton, William Enck, and Cristina Nita-Rotaru, editors, *IEEE Symposium on Security and Privacy, SP 2025, San Francisco, CA, USA, May 12-15, 2025*, pages 2583–2601. IEEE, 2025.

CMCP17. Alain Couvreur, Irene Márquez-Corbella, and Ruud Pellikaan. Cryptanalysis of McEliece cryptosystem based on algebraic geometry codes and their subcodes. *IEEE Transactions on Information Theory*, 63(8):5404–5418, 2017.

CMT23. Alain Couvreur, Rocco Mora, and Jean-Pierre Tillich. A new approach based on quadratic forms to attack the McEliece cryptosystem. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 3–38. Springer, 2023.

COT16. Alain Couvreur, Ayoub Otmani, and Jean-Pierre Tillich. Polynomial time attack on wild McEliece over quadratic extensions. *IEEE Transactions on Information Theory*, 63(1):404–427, 2016.

COTGU15. Alain Couvreur, Ayoub Otmani, Jean-Pierre Tillich, and Valérie Gauthier-Umana. A polynomial-time attack on the BBCRS scheme. In *IACR International Workshop on Public Key Cryptography*, pages 175–193. Springer, 2015.

CS98. Ronald Cramer and Victor Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In Hugo Krawczyk, editor, *Advances in Cryptology – CRYPTO'98*, volume 1462 of *Lecture Notes in Computer Science*, pages 13–25, Santa Barbara, CA, USA, August 23–27, 1998. Springer Berlin Heidelberg, Germany.

CS02. Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 45–64, Amsterdam, The Netherlands, April 28 – May 2, 2002. Springer Berlin Heidelberg, Germany.

DDMN12. Nico Döttling, Rafael Dowsley, Jörn Müller-Quade, and Anderson C. A. Nascimento. A CCA2 secure variant of the McEliece cryptosystem. *IEEE Trans. Inf. Theory*, 58(10):6672–6680, 2012.

DDN91. Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography (extended abstract). In *23rd Annual ACM Symposium on Theory of Computing*, pages 542–552, New Orleans, LA, USA, May 6–8, 1991. ACM Press.

DGHM18. Nico Döttling, Sanjam Garg, Mohammad Hajiabadi, and Daniel Masny. New constructions of identity-based and key-dependent message secure encryption schemes. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018: 21st International Conference on Theory and Practice of Public Key Cryptography,*

Part I, volume 10769 of *Lecture Notes in Computer Science*, pages 3–31, Rio de Janeiro, Brazil, March 25–29, 2018. Springer, Cham, Switzerland.

DMN12. Nico Döttling, Jörn Müller-Quade, and Anderson C. A. Nascimento. IND-CCA secure cryptography based on a variant of the LPN problem. In Xiaoyun Wang and Kazue Sako, editors, *Advances in Cryptology – ASIACRYPT 2012*, volume 7658 of *Lecture Notes in Computer Science*, pages 485–503, Beijing, China, December 2–6, 2012. Springer Berlin Heidelberg, Germany.

DMR25. Nico Döttling, Anne Müller, and Mahesh Sreekumar Rajasree. Separating pseudorandom codes from local oracles. Cryptology ePrint Archive, Paper 2025/1020, 2025.

DNR04a. Cynthia Dwork, Moni Naor, and Omer Reingold. Immunizing encryption schemes from decryption errors. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 342–360, Interlaken, Switzerland, May 2–6, 2004. Springer Berlin Heidelberg, Germany.

DNR04b. Cynthia Dwork, Moni Naor, and Omer Reingold. Immunizing encryption schemes from decryption errors. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 342–360. Springer, 2004.

Döt15. Nico Döttling. Low noise LPN: KDM secure public key encryption and sample amplification. In Jonathan Katz, editor, *PKC 2015: 18th International Conference on Theory and Practice of Public Key Cryptography*, volume 9020 of *Lecture Notes in Computer Science*, pages 604–626, Gaithersburg, MD, USA, March 30 – April 1, 2015. Springer Berlin Heidelberg, Germany.

FO99. Eiichiro Fujisaki and Tatsuaki Okamoto. How to enhance the security of public-key encryption at minimum cost. In Hideki Imai and Yuliang Zheng, editors, *PKC'99: 2nd International Workshop on Theory and Practice in Public Key Cryptography*, volume 1560 of *Lecture Notes in Computer Science*, pages 53–68, Kamakura, Japan, March 1–3, 1999. Springer Berlin Heidelberg, Germany.

FOP+16. Jean-Charles Faugere, Ayoub Otmani, Ludovic Perret, Frédéric De Portzamparc, and Jean-Pierre Tillich. Structural cryptanalysis of McEliece schemes with compact keys. *Designs, Codes and Cryptography*, 79(1):87–112, 2016.

FOPT10. Jean-Charles Faugere, Ayoub Otmani, Ludovic Perret, and Jean-Pierre Tillich. Algebraic cryptanalysis of McEliece variants with compact keys. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 279–298. Springer, 2010.

FS90. Uriel Feige and Adi Shamir. Witness indistinguishable and witness hiding protocols. In *22nd Annual ACM Symposium on Theory of Computing*, pages 416–426, Baltimore, MD, USA, May 14–16, 1990. ACM Press.

GG24. Surendra Ghentiyala and Venkatesan Guruswami. New constructions of pseudorandom codes. *arXiv preprint arXiv:2409.07580*, 2024.

GGW25. Sanjam Garg, Sam Gunn, and Mingyuan Wang. Black-box crypto is useless for pseudorandom codes. Cryptology ePrint Archive, Paper 2025/1021, 2025.

GK03. Shafi Goldwasser and Yael Tauman Kalai. On the (in)security of the Fiat-Shamir paradigm. In *44th Annual Symposium on Foundations of Computer Science*, pages 102–115, Cambridge, MA, USA, October 11–14, 2003. IEEE Computer Society Press.

GM84. Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.

GM24. Noah Golowich and Ankur Moitra. Edit distance robust watermarks for language models. *arXiv preprint arXiv:2406.02633*, 2024.

GOT12. Valérie Gauthier, Ayoub Otmani, and Jean-Pierre Tillich. A distinguisher-based attack on a variant of McEliece's cryptosystem based on Reed-Solomon codes. *arXiv preprint arXiv:1204.6459*, 2012.

GS95. Arnaldo Garcia and Henning Stichtenoth. A tower of Artin–Schreier extensions of function fields attaining the Drinfeld–Vladut bound. *Inventiones Mathematicae*, 121(1):211–222, 1995.

HKW20. Susan Hohenberger, Venkata Koppula, and Brent Waters. Chosen ciphertext security from injective trapdoor functions. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology – CRYPTO 2020, Part I*, volume 12170 of *Lecture Notes in Computer Science*, pages 836–866, Santa Barbara, CA, USA, August 17–21, 2020. Springer, Cham, Switzerland.

JST21. Fernando Granha Jeronimo, Shashank Srivastava, and Madhur Tulsiani. Near-linear time decoding of Ta-Shma's codes via splittable regularity. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 1527–1536, 2021.

JSV24. Nikola Jovanovic, Robin Staab, and Martin T. Vechev. Watermark stealing in large language models. In Ruslan Salakhutdinov, Zico Kolter, Katherine A. Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett,

and Felix Berkenkamp, editors, *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pages 22570–22593. PMLR / OpenReview.net, 2024.

Kil06. Eike Kiltz. Chosen-ciphertext security from tag-based encryption. In Shai Halevi and Tal Rabin, editors, *TCC 2006: 3rd Theory of Cryptography Conference*, volume 3876 of *Lecture Notes in Computer Science*, pages 581–600, New York, NY, USA, March 4–7, 2006. Springer Berlin Heidelberg, Germany.

KM23. Elena Kirshanova and Alexander May. Breaking Goppa-based McEliece with hints. *Information and Computation*, 293:105045, 2023.

KM25. Fuyuki Kitagawa and Takahiro Matsuda. Adaptive TDF from any TDF via pseudorandom-ciphertext PKE. *IACR Commun. Cryptol.*, 2(3):34, 2025.

KMP14. Eike Kiltz, Daniel Masny, and Krzysztof Pietrzak. Simple chosen-ciphertext security from low-noise LPN. In Hugo Krawczyk, editor, *PKC 2014: 17th International Conference on Theory and Practice of Public Key Cryptography*, volume 8383 of *Lecture Notes in Computer Science*, pages 1–18, Buenos Aires, Argentina, March 26–28, 2014. Springer Berlin Heidelberg, Germany.

KMT19. Fuyuki Kitagawa, Takahiro Matsuda, and Keisuke Tanaka. CCA security and trapdoor functions via key-dependent-message security. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 33–64, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Cham, Switzerland.

KW19. Venkata Koppula and Brent Waters. Realizing chosen ciphertext security generically in attribute-based encryption and predicate encryption. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part II*, volume 11693 of *Lecture Notes in Computer Science*, pages 671–700, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Cham, Switzerland.

Lam79. Leslie Lamport. Constructing Digital Signatures from a One Way Function. October 1979.

LN09. Gaëtan Leurent and Phong Q. Nguyen. How risky is the random-oracle model? In Shai Halevi, editor, *Advances in Cryptology – CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 445–464, Santa Barbara, CA, USA, August 16–20, 2009. Springer Berlin Heidelberg, Germany.

LQR+19. Alex Lombardi, Willy Quach, Ron D. Rothblum, Daniel Wichs, and David J. Wu. New constructions of reusable designated-verifier NIZKs. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 670–700, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Cham, Switzerland.

LS02. Pierre Loidreau and Nicolas Sendrier. Weak keys in the McEliece public-key cryptosystem. *IEEE Transactions on Information Theory*, 47(3):1207–1211, 2002.

McE78. Robert J McEliece. A public-key cryptosystem based on algebraic. *Coding Thv*, 4244(1978):114–116, 1978.

Mce26. Classic McEliece. `https://classic.mceliece.org/papers.html`, 2026. [Accessed 12-02-2026].

MP12. Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 700–718, Cambridge, UK, April 15–19, 2012. Springer Berlin Heidelberg, Germany.

Nao90. Moni Naor. Bit commitment using pseudo-randomness. In Gilles Brassard, editor, *Advances in Cryptology – CRYPTO'89*, volume 435 of *Lecture Notes in Computer Science*, pages 128–136, Santa Barbara, CA, USA, August 20–24, 1990. Springer, New York, USA.

Nie86. Harald Niederreiter. Knapsack-type cryptosystems and algebraic coding theory. *Prob. Contr. Inform. Theory*, 15(2):157–166, 1986.

Nie02. Jesper Buus Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In Moti Yung, editor, *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 111–126, Santa Barbara, CA, USA, August 18–22, 2002. Springer Berlin Heidelberg, Germany.

NY90. Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *22nd Annual ACM Symposium on Theory of Computing*, pages 427–437, Baltimore, MD, USA, May 14–16, 1990. ACM Press.

PHZS24. Qi Pang, Shengyuan Hu, Wenting Zheng, and Virginia Smith. No free lunch in llm watermarking: Trade-offs in watermarking design choices, 2024.

PW08. Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In Richard E. Ladner and Cynthia Dwork, editors, *40th Annual ACM Symposium on Theory of Computing*, pages 187–196, Victoria, BC, Canada, May 17–20, 2008. ACM Press.

QRW19. Willy Quach, Ron D. Rothblum, and Daniel Wichs. Reusable designated-verifier NIZKs for all NP from CDH. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2019, Part II*, volume 11477 of *Lecture Notes in Computer Science*, pages 593–621, Darmstadt, Germany, May 19–23, 2019. Springer, Cham, Switzerland.

Ran25. Hugues Randriambololona. The syzygy distinguisher. In Serge Fehr and Pierre-Alain Fouque, editors, *Advances in Cryptology - EUROCRYPT 2025 - 44th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Madrid, Spain, May 4-8, 2025, Proceedings, Part VI*, volume 15606 of *Lecture Notes in Computer Science*, pages 324–354. Springer, 2025.

Sah99. Amit Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *40th Annual Symposium on Foundations of Computer Science*, pages 543–553, New York, NY, USA, October 17–19, 1999. IEEE Computer Society Press.

SKS19. Jad Silbak, Swastik Kopparty, and Ronen Shaltiel. Quasilinear time list-decodable codes for space bounded channels. In David Zuckerman, editor, *60th Annual Symposium on Foundations of Computer Science*, pages 302–333, Baltimore, MD, USA, November 9–12, 2019. IEEE Computer Society Press.

SS92. Vladimir Michilovich Sidelnikov and Sergey O Shestakov. On insecurity of cryptosystems based on generalized Reed-Solomon codes. 1992.

Sti09. Henning Stichtenoth. *Algebraic Function Fields and Codes*, volume 254 of *Graduate Texts in Mathematics*. Springer, 2 edition, 2009.

vT94. Johan van Tilburg. Security-analysis of a class of cryptosystems based on linear error-correcting codes. 1994.

Xag22. Keita Xagawa. The boneh-katz transformation, revisited: Pseudorandom/obliviously-samplable PKE from lattices and codes and its application. In Riham AlTawy and Andreas Hülsing, editors, *SAC 2021: 28th Annual International Workshop on Selected Areas in Cryptography*, volume 13203 of *Lecture Notes in Computer Science*, pages 47–67, Virtual Event, September 29 – October 1, 2022. Springer, Cham, Switzerland.

Zha22. Mark Zhandry. Augmented random oracles. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology – CRYPTO 2022, Part III*, volume 13509 of *Lecture Notes in Computer Science*, pages 35–65, Santa Barbara, CA, USA, August 15–18, 2022. Springer, Cham, Switzerland.