Error-Correcting Codes against Chosen-Codeword Attacks

Kenji Yasunaga

Kanazawa University, Kakuma-machi Kanazawa, Japan
yasunaga@se.kanazawa-u.ac.jp

Abstract. We study the problem of error correction for computationally bounded channels under chosen-codeword attacks (CCA). In the CCA setting, the channel can introduce a p-fraction of errors by accessing to the encoding and the decoding oracles. Since the unique decoding is not possible for $p \geq 1/4$, we consider list-decodable codes. We present an optimal-rate coding scheme by assuming the existence of one-way functions. The construction is based on the list-decodable code of Guruswami and Smith (2010) for computationally bounded channels.

1 Introduction

The problem of error correction for computationally bounded channels was first studied by Lipton [9]. He reduced the problem of error correction in the secret-key (or shared randomness) setting to the problem of error correction in binary symmetric channels (BSC). In his scheme, a message m is encoded into $c = \pi^{-1}(C(m)) \oplus \mu$, where C is an error correcting code for BSC, and π and μ are a random bit permutation and a random mask, respectively, which are privately shared between the sender and the receiver. When an error e was introduced, on input $y = c \oplus e$, the decoder sends $\pi(y \oplus \mu) = C(m) \oplus \pi(e)$ to the decoder of C. Since $\pi(e)$ is like a random error in BSC, C can correct it. The scheme can achieve an optimal rate in the secret-key setting since there are several optimal-rate (capacity-achieving) schemes for BSC [2, 1]. Lipton showed that, by assuming a pseudorandom generator exists, the length of the secret key can be reduced to n^{-d} for any constant $d > 0$, where n is the length of c.

Lipton's scheme beautifully reduced the error correction in the secret-key setting to that in BSC. However, the scheme only achieves *one-time* security. Thus, in order to send k messages reliably, we need to generate k secret keys. The situation is very similar to that of the one-time pad for secret-key encryption.

In modern cryptography, more powerful attacks are considered for encryption schemes, for which the one-time pad is not secure. In a chosen-plaintext attack (CPA), an adversary is allowed to access to the encryption oracle. In addition, the adversary can access to the decryption oracle in a chosen-ciphertext attack (CCA). The CCA security is widely accepted as the standard security notion for encryption schemes.

In this work, we introduce an analogous notion to chosen-ciphertext attack security, called *chosen-codeword attack (CCA)* security, for the problem of error

correction. Intuitively, a coding scheme is said to be CCA secure if for any probabilistic polynomial-time (PPT) adversarial channel, the scheme can correct any error introduced by the channel even if it can access to both the encoding and the decoding oracles. We assume that adversarial channels can introduce at most p-fraction errors, where $p \in (0, 1/2)$ is called the error rate. The CCA security of the coding scheme captures strong attack scenarios, in which adversarial channels can introduce errors by using various information obtained via the encoding and decoding functions.

We present an optimal-rate coding scheme that is CCA secure in the secret-key setting. The construction is based on the framework of Guruswami and Smith [6], which provides a one-time secure coding scheme for computationally bounded channels.

It is first observed that, in order to tolerate attacks using the encoding oracle, it is necessary to relax the goal of decoding to *list decoding*, where the decoder outputs a polynomial-size list containing an original message. As discussed in [10], if an adversarial channel can obtain polynomially-many valid codewords, the channel can cause a non-negligible error for unique decoding when the error rate $p > 1/4$. Thus, we aim to construct a list-decodable code with optimal rate $1 - H(p) - \varepsilon$ for any constant ε, where p is the error rate and $H(\cdot)$ is the binary entropy function.

Guruswami and Smith [6] showed an optimal-rate list-decodable code for computationally bounded channels. Their scheme does not assume the existence of a shared secret key. However, they only presented a probabilistic construction that is not fully explicit. To implement their scheme, the sender and the receiver need to share the random coins privately to the channel. The only probabilistic part of [6] is a primitive called a *pseudorandom code*, which is a list-decodable code whose codewords themselves are pseudorandom. We would like to construct an explicit scheme that is secure against polynomial-time channels. Since the scheme of [6] can been seen as an explicit scheme in the secret-key setting. we aim to construct an explicit secret-key code that is CCA secure for polynomial-time adversaries.

Our scheme is built with several cryptographic primitive including pseudorandom generators, pseudorandom functions, and secure message authentication codes. All the three primitives can be constructed by assuming the existence of one-way functions [3, 4, 7].

1.1 Ideas of the Construction

Our approach is to enhance the error correctability of the scheme of [6] to have CCA security. In order to prove CCA security, we need to show that the responses of the encoding/decoding oracles are useless for an adversarial channel. It is sufficient to show that the oracle responses can be *simulated* without using the secret key.

We observe that the codewords of the scheme [5] are pseudorandom. Thus, simulation of the encoding oracle can be done by preparing a random string for every oracle query. When an adversarial channel makes the i-th query m_i

to the encoding oracle, the simulator responds with a random string c_i. By pseudorandomness of codewords, the simulation can be done successfully.

To achieve CCA security, we need to simulate the decoding oracle. As in achieving CCA security for encryption schemes, we employ a message authentication code (MAC). To encode a message m, we first generate a MAC tag τ of m, and feed (m, τ) to the encoder. By this modification, the adversarial channel cannot generate a valid codeword without querying the encoding oracle. Otherwise, such a channel can be used for breaking the security of MAC. As in simulating the encoding oracle, we use a list of pairs (m_i, c_i) for every oracle query m_i to the encoding oracle, where c_i is a random string. On querying y to the decoding oracle, the simulator responds with the list of all m_i for which the corresponding codeword c_i is within a distance $\lceil pn \rceil$ from y. Since the encoding of [6] is inherently probabilistic, it seems necessary to check exponentially-many possible codewords of m_i for each i. However, this problem can be resolved by making the encoding *deterministic*, which can be done by using a pseudorandom function F. In encoding a message m, the value $F_{sk}(m)$ is used as random coins for the encoder, where sk is the secret key of F shared between the sender and the receiver. As long as sk is secret, $F_{sk}(m)$ looks random. Since there is a unique codeword for each message m_i, the responses of the decoding oracle can be simulated in the above way.

We employ several cryptographic primitives, and hence our construction is relatively simple compared to the original construction of [6]. We could avoid using a randomness-efficient sampler, a generator of t-wise independent permutations, a generator of t-wise independent strings, and a code correcting t-wise independent errors. Instead, we use standard cryptographic primitives, a pseudorandom generator, a pseudorandom function, and a message authentication code.

1.2 Related Work

Lipton [9] introduced the notion of computationally bounded channels, and proposed an optimal-rate scheme that has one-time security in the secret-key setting.

Langberg [8] studied secret-key coding schemes for computationally *unbounded* channels, and showed the existence of an optimal-rate coding scheme with optimal secret-key size.

Micali et al. [10, 11] proposed a coding scheme in the public-key setting in which the encoder has a secret key of digital signature and the decoder has the corresponding verification key. They considered an attack scenario in which an adversarial channel can observe several valid codewords. However, their security model is essentially different from the model in which the channel can access to the encoding oracle. In the model of [10, 11], the sender can use the *time stamp* to encode messages. Since the channel is not allowed to access to the signing oracle, it cannot obtain multiple codewords with the same time stamp. Indeed, the use of time stamps allows to circumvent the impossibility of unique decoding.

Guruswami and Smith [6] studied coding schemes for computationally bounded channels where the time complexity is a prior bounded polynomial.

They presented a probabilistic construction of an optimal-rate coding scheme in which neither a secret key nor a public key is used.

2 Preliminaries

2.1 Notations

For $n \in \mathbb{N}$, we write $[n] = \{1, 2, \ldots, n\}$. For a finite set Σ and an integer $n \in \mathbb{N}$, $x \in \Sigma^n$ is called a vector or string over the alphabet Σ. A function $f : \mathbb{N} \to \mathbb{N}$ is called *negligible* in n if for every positive polynomial $p(\cdot)$, there exists $n_0 \in \mathbb{N}$ such that for every integer $n > n_0$, $f(n) < 1/p(n)$. We write $\mathsf{negl}(\cdot)$ as a negligible function. The uniform distribution over $\{0,1\}^n$ is denoted by U_n.

2.2 Error-Correcting Codes

The Hamming distance between two vectors $x, y \in \Sigma^n$ is defined to be $\Delta(x, y) = |\{i \in [n] : x_i \neq y_i\}|$, where $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$. The Hamming weight of $x \in \Sigma^n$ is $\mathsf{wt}(x) = |\{i \in [n] : x_i \neq 0\}|$. For a finite set Σ and $R \in (0, 1]$, a *code* over an alphabet Σ with an *information rate* R is a mapping $C : \Sigma^k \to \Sigma^n$, where $k/n = R$. We refer to k and n as the *message length* and the *code length*, respectively. An encoded message $C(x)$ is called a *codeword*. We also refer to C as the set $\{C(x) : x \in \Sigma^k\}$ of the codewords.

A code C over Σ is called (δ, L)*-list decodable* if for any $y \in \Sigma^n$, there is at most $\ell \leq L$ codewords $c_1, \ldots, c_\ell \in C$ such that $\Delta(y, c_i) \leq \delta n$ for all $i \in [\ell]$.

3 Formal Model

We define a coding scheme that is parameterized by a security parameter n, which is equal to the code length.

Definition 1 (Secret-Key Coding Scheme). *A* secret-key coding scheme Π *consists of three polynomial-time algorithms* Setup, Enc, Dec, *a finite alphabet* Σ, *and an information rate* R. *For every sufficiently large* $k \in \mathbb{N}$, *let* $n = \lfloor k/R \rfloor$. *On input* 1^n, Setup *outputs the secret key* sk. *On input* 1^n, sk, *and a message* $m \in \Sigma^k$, Enc *outputs a codeword* $c \in \Sigma^n$. *On input* 1^n, pk, *and a string* $y \in \Sigma^n$, Dec *outputs a message* $m \in \Sigma^k$ *or* $\perp$. *The input of the parameter* 1^n *to* Enc *and* Dec *may be omitted for simplicity. It is required that for every* k *and key* sk *generated by* $\mathsf{Setup}(1^n)$, *it holds that* $\Pr[\mathsf{Dec}_{sk}(\mathsf{Enc}_{sk}(m)) = m] = 1$ *for any* $m \in \{0,1\}^k$.

An adversarial channel W consists of two algorithms W_1 and W_2. To define the error correctability, we define the following game for a coding scheme over Σ of rate R. First, the setup algorithm generates the secret key. Then, W_1 chooses a challenge message $m^* \in \Sigma^k$. After that, on input the codeword c^* of the chosen message, W_2 outputs $y \in \Sigma^n$, where $n = \lfloor k/R \rfloor$. A channel is called p*-error* if

W_2, on input c^*, always outputs y satisfying $\Delta(c^*, y) \leq \lceil pn \rceil$. We say that W is *probabilistic polynomial-time (PPT)* if both W_1 and W_2 run in time polynomial in n.

We define the security against chosen-codeword attack (CCA security). In this security, the encoding oracle $\mathsf{Enc}_{sk}(\cdot)$ and the decoding oracle $\mathsf{Dec}_{sk}(\cdot)$ are available to an adversarial channel. Since the unique decoding is not possible if the channel can access to the encoding oracle and the error rate $p > 1/4$, we say that a coding scheme is CCA secure if the scheme can output a polynomial-size list that contains the challenge message.

Definition 2 (CCA Security). *Let $\Pi = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec}, \Sigma, R)$ be a secret-key coding scheme of rate R, and $W = (W_1, W_2)$ an adversarial channel. For $k \in \mathbb{N}$, we define the advantages of W as*

$$\mathsf{Adv}^{CCA}_{W,\Pi}(k) = \Pr\left[m^* \notin L : \begin{array}{l} sk \leftarrow \mathsf{Setup}(1^n), (m^*, st) \leftarrow W_1^{\mathsf{Enc}_{sk}(\cdot),\mathsf{Dec}_{sk}(\cdot)}(1^n), \\ c^* \leftarrow \mathsf{Enc}_{sk}(m^*), y^* \leftarrow W_2^{\mathsf{Enc}_{sk}(\cdot),\mathsf{Dec}_{sk}(\cdot)}(st, c^*), \\ L \leftarrow \mathsf{Dec}_{sk}(y^*) \end{array} \right],$$

*where $n = \lfloor k/R \rfloor$. We say Π is (p, T, ε)-*CCA secure *if $\mathsf{Adv}^{CCA}_{W,\Pi}(k) \leq \varepsilon(n)$ for every p-error probabilistic $T(n)$-time channel W.*

4 Our Construction

4.1 Overview

Our construction is based on the framework of Guruswami and Smith [6].

A message m is encoded as $\pi^{-1}(C(m)) \oplus \mu$, where C is a random-error correcting code, π is a random bit permutation, and μ is a random mask. This part is called a *payload* codeword, and is divided into $\ell - \kappa$ blocks $C_1, \ldots, C_{\ell-\kappa}$. The seeds s_π and s_μ for π and μ should be shared between the sender and the receiver. We will not include s_π and s_μ in the secret key. Instead, we will send them privately by jamming them into the payload codeword. The control information s is encoded by a Reed-Solomon code, so that it will be recovered by a (generalized) list decoding. Let $(f(\alpha_1), \ldots, f(\alpha_\kappa))$ be the encoded information, where each α_i is an element of a finite field, and f is the polynomial corresponding to the control information s. Then, each pair of element $(\alpha_i, f(\alpha_i))$ is encoded into a block D_i by a *pseudorandom code (PRC)*. The resulting blocks $D_1, \ldots, D_\kappa$ are randomly mixed into the payload blocks $(C_1, \ldots, C_{\ell-\kappa})$. The information V of the positions of the control-information blocks is also included into the control information s. By the property of PRC, each block C_i is pseudorandom. Since the payload codeword is masked by a random string μ, the resulting codeword is also pseudorandom.

In decoding, first, the list decoding of the pseudorandom code is applied to each block. Then, a list of pairs $(\alpha_j, \beta_{jh})_h$ is recovered for each j. By applying the

(generalized) list decoding of the Reed-Solomon code to the list $\{(\alpha_j, \beta_{jh})_h\}_j$, we can recover a list of control information. For each candidate control information s, by recovering π, μ, V from s, the payload part is decoded with the decoder of the random-error correcting code.

As described in Section 1.1, we use pseudorandom functions (PRF) to make the encoding deterministic. The first PRF is used for generating the control information from a message m. The second PRF is for sampling random coins for the pseudorandom code. Also, as explained in Section 1.1, we add a MAC tag to the message m. The secret key consists of the two keys of PRFs and the secret keys of MAC and PRC.

4.2 Ingredients

We use the following tools in our construction:

- A random p-error correcting code $\mathsf{REC} : \{0,1\}^{R'n} \to \{0,1\}^n$ of rate $R' = 1-H(p)-\varepsilon'$ for any positive constant ε'. It is required that for every message $m \in \{0,1\}^{R'n}$ and error vector $e \in \{0,1\}^n$ of Hamming weight at most $\lceil pn \rceil$, the decoder of REC, on input $\mathsf{REC}(m) \oplus \pi(e)$, outputs m with probability at least $1 - \mathsf{negl}(n)$, where π is a random bit permutation. Any capacity-achieving code in binary symmetric channels with cross-over probability p satisfies the property.
- A Reed-Solomon code $\mathsf{RS} : \mathbb{F}^{k+1} \to \mathbb{F}^n$ of rate $R_1 = O(\varepsilon)$ that enables a generalized list decoding. For n distinct elements $\{\alpha_1, \ldots, \alpha_n\}$ from a finite field $\mathbb{F}$, the codeword of $m = (m_0, \ldots, m_k) \in \mathbb{F}^{k+1}$ is $\mathsf{RS}(m) = (f(\alpha_1), \ldots, f(\alpha_n))$, where $f(X) = m_0 + m_1 X + \cdots + m_k X^k$. The list decoding property guarantees that, given n distinct pairs $(\alpha_i, \beta_i) \in \mathbb{F}^2$ for $i \in [n]$, one can find a list P of all polynomials f of degree at most k that satisfy $f(\alpha_i) = \beta_i$ for at least t values of $i \in [n]$. Sudan's algorithm [12] is sufficient for our purpose. It runs in time polynomial in n and $\log |\mathbb{F}|$, and works as long as the agreement parameter $t > \sqrt{2kn}$. The size of the list P is at most $\sqrt{2n/k}$.
- A pseudorandom code family $\mathsf{PRC} = \{\mathsf{PRC}_s : \{0,1\}^{R_2 b} \times \{0,1\}^b \to \{0,1\}^b\}$ of rate R_2 indexed by a key $s \in \{0,1\}^b$ with the following properties: (1) For any constant $\varepsilon < 1/2$, PRC_s is $(1/2-\varepsilon, L)$-list decodable with high probability, where L and R_2 only depend on ε, independent of b, and the probability is taken over $s \in \{0,1\}^b$; (2) For any $m \in \{0,1\}^{R_2 b}$, $\mathsf{PRC}_s(m, U_b)$ is pseudorandom. Specifically, it is required that for any n^c-time adversary and a sequence of $q = n^d$ messages $(m_1, \ldots, m_q) \in (\{0,1\}^{R_2 b})^q$, it is difficult to distinguish $(\mathsf{PRC}(m_1, r_1), \ldots, \mathsf{PRC}(m_\ell, r_q))$ from U_{qb} with probability more than $1/n^c$, where $c, d > 0$ are constants, and each r_i is chosen uniformly at random from $\{0,1\}^b$.
  A probabilistic construction with parameters $R_2 \geq \varepsilon^{O(1)}$ and $L \leq (1/\varepsilon)^{O(1)}$ is presented in [6]. Their construction can be seen as an explicit construction in the secret-key setting, where a shared key s is used for choosing a pseudorandom generator for n^c-time adversaries. By setting $b = O(\log n)$, the list-decoding algorithm can be performed in time polynomial in n. We use this explicit construction in our scheme.

– A pseudorandom generator (PRG) $G : \{0,1\}^n \to \{0,1\}^{p(n)}$, where $p(\cdot)$ is any polynomial. It is required that for any PPT algorithm, it is difficult to distinguish $G(U_n)$ from the uniform distribution $U_{p(n)}$. Such G exists assuming one-way functions exist [7].
– A pseudorandom function (PRF) family $\mathcal{F} = \{F^n\}_{n\in\mathbb{N}}$, where $F^n = \{F_s : \{0,1\}^n \to \{0,1\}^n\}_{s\in\{0,1\}^n}$ is a collection of functions indexed by a key $s \in \{0,1\}^n$. For any PPT algorithm, it is difficult to distinguish whether it has oracle access to F_s or a random function. Such $\mathcal{F}$ can be constructed assuming the existence of one-way functions [4, 7].
– A message authentication code (MAC) $(\mathsf{Tag}, \mathsf{Vrfy})$, where the key is chosen uniformly at random from $\{0,1\}^n$. It guarantees that for any PPT adversary, given access to Tag and Vrfy oracles, it is difficult to forge a pair (m,t) of a message and a tag that passes the verification of Vrfy. We use a code with short tag. Namely, on input a key $s \in \{0,1\}^n$ and a message $m \in \{0,1\}^n$, Tag outputs a tag $t \in \{0,1\}^{n^\gamma}$ for some constant $\gamma \in (0,1)$. Such a code exists assuming the existence of one-way functions [3, 4, 7].
– A generator P for permutations $\pi : [n] \to [n]$ of the set $[n]$ that uses $O(n \log n)$ bits to specify a permutation. We use a straightforward construction in which a permutation is chosen from the set of the possible $n!$ permutations of $[n]$. For a vector $x = (x_1, \ldots, x_n) \in \{0,1\}^n$, we write $\pi(x) = (y_1, \ldots, y_n)$, where $y_{\pi(i)} = x_i$ for $i \in [n]$.

4.3 The Construction

For any positive real $p < 1/2$ and ε, we construct a code of rate $R = 1 - H(p) - \varepsilon$ that is p-CCA secure. We assume that $2\varepsilon < 1/2 - p$. For a message length $k \in \mathbb{N}$, the code length is $n = \lfloor k/R \rfloor$. For any message $m \in \{0,1\}^k$, the encoded codeword $c \in \{0,1\}^n$ consists of ℓ blocks $(B_1, \ldots, B_\ell)$, where each block B_i is of length $b = c \log n$ for some constant $c > 0$, and thus $\ell = n/(c \log n)$. We set the following parameters: $\lambda = n^\gamma$ for a constant $\gamma \in (0,1)$, $n' = \lfloor (k+\lambda)/R' \rfloor = (\ell - \kappa)b$, $\kappa = \lceil 6\lambda/(R_1 R_2 b) \rceil$, where R' is the rate of REC, $R_1 = O(\varepsilon^2/L^2)$ is the rate of RS over $\mathbb{F}$ with $|\mathbb{F}| = 2^{R_2 b/2}$, and $R_2 = \varepsilon^{O(1)}$ is the rate of PRC.

Setup Algorithm. On input a parameter 1^n, the setup algorithm chooses four random keys $s_1, s_2, s_3, s_4 \in \{0,1\}^n$ for two PRF families $\mathcal{F}_1$ and $\mathcal{F}_2$, a MAC $(\mathsf{Tag}, \mathsf{Vrfy})$, and a PRC family PRC, respectively. The first PRF $\mathcal{F}_1$ consists of $\{F_s^1 : \{0,1\}^k \to \{0,1\}^{3\lambda}\}_{s\in\{0,1\}^n}$. The second PRF $\mathcal{F}_2$ consists of $\{F_s^2 : \{0,1\}^k \times [\ell] \to \{0,1\}^b\}_{s\in\{0,1\}^n}$. The tagging algorithm Tag, on input a secret key s_3 and a message $m \in \{0,1\}^k$, outputs a tag of length λ. The secret key SK is (s_1, s_2, s_3, s_4).

Encoding Algorithm. The encoding consists of the payload encoding and the control-information encoding. On input a message $m \in \{0,1\}^k$, the control information $s = (s_\pi, s_\mu, s_V) \in \{0,1\}^{3\lambda}$ is generated as $F_{s_1}^1(m)$, where $|s_\pi| = |s_\mu| = |s_V| = \lambda$.

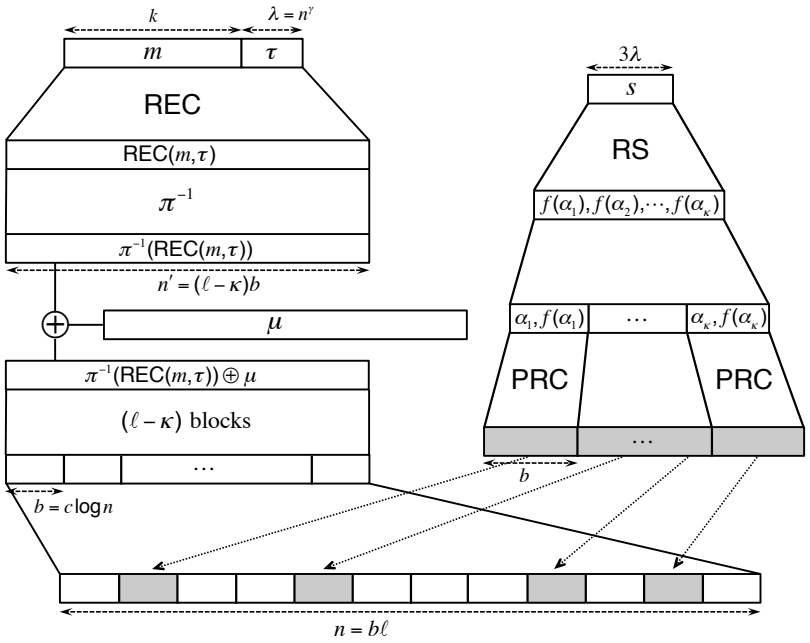

Fig. 1. Construction of the Code

The payload codeword is generated as follows: first, the tag $\tau = \mathsf{Tag}_{s_3}(m)$ is generated, and a PRG $G_\pi : \{0,1\}^\lambda \to \{0,1\}^{O(n \log n)}$ is used to produce $r = G_\pi(s_\pi)$. We use r and P (the generator of permutation of the set $[n']$) to generate a permutation $\pi = P(r)$. A PRG $G_\mu : \{0,1\}^\lambda \to \{0,1\}^{n'}$ is used to generate $\mu = G_\mu(s_\mu)$. Finally, we take π and $\mathsf{REC} : \{0,1\}^{k+\lambda} \to \{0,1\}^{n'}$ to produce the payload codeword $c_p = \pi^{-1}(\mathsf{REC}(m,\tau)) \oplus \mu$.

In the control-information encoding, first, s is encoded to a codeword $(f(\alpha_1), \ldots, f(\alpha_\kappa))$ by Reed-Solomon code RS of rate R_1 over a finite field $\mathbb{F}$. A PRG $G_V : \{0,1\}^\lambda \to [\ell]^\kappa$ is used to generate $r' = G_V(s_V)$. We use r' to produce a set $V = \{v_1, \ldots, v_\kappa\}$ of distinct κ random samples in $[\ell]$. Let $\mathsf{PRC} : \mathbb{F}^2 \times \{0,1\}^b \to \{0,1\}^b$ be a pseudorandom code of rate R_2 that is $(p+\varepsilon, L)$-list decodable, where $p + \varepsilon < 1/2 - \varepsilon$, $R_2 = \varepsilon^{O(1)}$, and $L = (1/\varepsilon)^{O(1)}$. For $i \in [\kappa]$, each pair of elements $(\alpha_i, f(\alpha_i))$ is encoded to the v_i-th block $B_{v_i} = \mathsf{PRC}((\alpha_i, f(\alpha_i)), F^2_{s_2}(m,i))$, where the output of PRF is used for random coins. The control-information codeword is $(B_{v_1}, B_{v_2}, \ldots, B_{v_\kappa})$.

The payload codeword c_p is divided into $\ell - \kappa$ blocks $(B_{i_1}, B_{i_2}, \ldots, B_{i_{\ell-\kappa}})$, where i_j is the j-th smallest element in $[\ell] \setminus V$, and each block $B_i \in \{0,1\}^b$ and $n' = (\ell - \kappa)b$. The final codeword is $(B_1, B_2, \ldots, B_\ell) \in (\{0,1\}^b)^\ell = \{0,1\}^n$.

The construction is summarized in Figure 1.

Decoding Algorithm. On input $y \in \{0,1\}^n$, divide y into ℓ blocks $(Y_1, Y_2, \ldots, Y_\ell) \in (\{0,1\}^b)^\ell$.

For $i \in [\ell]$, decode block Y_i by the list-decoding algorithm of PRC. By combining the output lists for all $i \in [\ell]$, a list L_1 of pairs $\{(\alpha_j, \beta_{jh})_h\}_j$ for $j \in [\kappa]$ is obtained. The size of L_1 is at most ℓL. Then, apply the list-decoding algorithm of RS to L_1 with an agreement parameter $t = \varepsilon\kappa/2$ to generate a list L_2 of control information $\tilde{s} = (\tilde{s}_\pi, \tilde{s}_\mu, \tilde{s}_V)$. The size of L_2 is at most $\sqrt{2/R_1} = O(L/\varepsilon)$.

Let L_3 be the empty list. For each $\tilde{s} = (\tilde{s}_\pi, \tilde{s}_\mu, \tilde{s}_V) \in L_2$, do the following. Recover $\tilde{\pi} = P(G_\pi(\tilde{s}_\pi))$, $\tilde{\mu} = G(\tilde{s}_\mu)$, and $\tilde{V} = \{v_1, \ldots, v_\kappa\}$, where $G_V(\tilde{s}_V)$ is used to produce $\tilde{V}$. Let $\tilde{y}$ be the concatenation of blocks $(Y_{i_1}, Y_{i_2}, \ldots, Y_{i_{\ell-\kappa}}) \in \{0,1\}^{n'}$, where i_j is the j-th smallest element in $[\ell] \setminus \tilde{V}$. Decode $\tilde{z} = \pi(\tilde{y} \oplus \tilde{\mu})$ with the decoding algorithm of REC. Let $(\tilde{m}, \tilde{\tau})$ be the output. If $\mathsf{Vrfy}(\tilde{m}, \tilde{\tau})$ outputs false, go back and choose next $\tilde{s}$ from L_2. Otherwise, recover $\tilde{s}'$ as $F_{s_1}(\tilde{m})$. If $\tilde{s}' \neq \tilde{s}$, go back and choose next $\tilde{s}$ from L_2. Else, recover the encoded message $\tilde{c}$ for a message $\tilde{m}$ and a control information $\tilde{s}$ by following the encoding algorithm. Check if the Hamming distance between $\tilde{c}$ and y is at most $\lceil pn \rceil$. If so, add $\tilde{m}$ to L_3; and otherwise do nothing. Then, go back and choose next $\tilde{s}$ from L_2.

After choosing all elements in L_2, output L_3 if $L_3 \neq \emptyset$, and $\perp$ otherwise.

The Rate of the Code. For any positive constant $p < 1/2$ and ε, we choose a random p-error correcting code of rate $R' = 1 - H(p) - \varepsilon'$ such that $0 < \varepsilon' < \varepsilon$. It holds that $R' = (k + \lambda)/n'$. The length of the control-information blocks is $\kappa b \leq c_0 b$ for sufficiently large constant c_0. Thus, the rate of the code is

$$R = \frac{k}{n' + \kappa b} \geq \frac{R'n' - \lambda}{n' + c_0 b} = R' - \frac{R'c_0 b/\lambda + 1}{(n' + c_0 b)/\lambda}.$$

Note that $b = c \log n$ and $\lambda = n^\gamma$ for $\gamma \in (0,1)$. Since $R'c_0 b/\lambda = o(1)$ and $(n' + c_0 b)/\lambda = \Omega(n^{1-\gamma})$, we have that $R = 1 - H(p) - \varepsilon' - O(n^{-(1-\gamma)}) \geq 1 - H(p) - \varepsilon$ for sufficiently large n.

5 Security Proof

We prove the following theorem.

Theorem 1. *Assume that there exist one-way functions. For any positive constants $p < 1/2, \varepsilon, c$, the coding scheme of rate $R \geq 1 - H(p) - \varepsilon$ described in Section 4.3 is (p, n^c, n^{-c})-CCA secure.*

Proof. We prove the security with a sequence of hybrid games where the first game corresponds to the original CCA security game.

- Game_0: The original CCA security game between the challenger and the channel.
  1. The secret key $SK = (s_1, s_2, s_3, s_4) \in \{0,1\}^{4n}$ is chosen uniformly at random.
  2. The channel makes queries for $m \in \{0,1\}^k$ to the encoding oracle and for $y \in \{0,1\}^n$ to the decoding oracle, and outputs $m^* \in \{0,1\}^k$. The responses of the oracles are made according to the encoding and the decoding algorithms.

3. Let $s = (s_\pi, s_\mu, s_V) = F^1_{s_1}(m^*)$. The payload codeword is generated as $\pi^{-1}(\mathsf{REC}(m, \tau)) \oplus \mu$, where $\pi = P(G_\pi(s_\pi))$, $\tau = \mathsf{Tag}_{s_3}(m^*)$, $\mu = G_\mu(s_\mu)$. Then, s is encoded as $(f(\alpha_1), \ldots, f(\alpha_\kappa))$ by RS. Let $V^* = \{v_1, \ldots, v_\kappa\}$ be the set generated from the seed $r' = G_V(s_V)$. The v_i-th block B_{v_i} is generated as $\mathsf{PRC}_{s_4}((\alpha_i, f(\alpha_i)), F^2_{s_2}(m, i))$. The payload codeword is divided into $\ell - \kappa$ blocks $(B_{i_1}, \ldots, B_{i_{\ell-\kappa}})$, where i_j is the j-th smallest element in $[\ell] \backslash V^*$. The challenge codeword $c^* = (B_1, B_2, \ldots, B_\ell)$ is given to the channel.
4. The channel makes queries to the encoding and the decoding oracles, and outputs $y^* \in \{0,1\}^n$ such that $\Delta(y, c^*) \leq \lceil pn \rceil$.
5. Let L be the output of the decoding algorithm on input y^*. If $m^* \notin L$, the channel wins. Otherwise, the channel loses.

- Game_1: The same as Game_0 except that all the outputs of PRFs are replaced with uniformly random strings. Specifically, s is chosen uniformly at random, and the uniformly random bits are used as random coins for PRC.
- Game_2: The same as Game_1 except that all the outputs of PRGs are replaced with uniformly random strings. Specifically, the input to the permutation generator P and $(v_1, \ldots, v_\kappa)$ are chosen uniformly at random.
- Game_3: The same as Game_2 except that for every query m to the encoding oracle and the challenge message m^*, the corresponding codeword is generated by choosing a uniformly random string from $\{0,1\}^n$.
- Game_4: The same as Game_3 except that for every query m to the encoding oracle, prepare an entry (m, c) in a list L_Q, where c is the codeword of m. The pair of the challenge message and codeword (m^*, c^*) is also included in L_Q. For every query m to the encoding oracle, responds with c such that $(m, c) \in L_Q$; for every query y to the decoding oracle, respond with a list $\{m \in \{0,1\}^k : (m, c) \in L_Q \text{ and } \Delta(y, c) \leq \lceil pn \rceil\}$.
- Game_5: The same as Game_4 except that the channel is not allowed to query the oracles at Steps 2 and 4.

The probability that a channel wins is called an advantage of the channel.

First, it can be shown that the advantage of any PPT channel in Game_1 is negligibly close to that in Game_0. This is because the keys s_1 and s_2 for PRFs are not disclosed to the channel, and thus the security of PRF guarantees it.

It follows from the security of PRG that the advantage of any PPT channel in Game_2 is negligibly close to that in Game_1.

In Game_3, all the codewords are chosen uniformly at random. The change is not detected by n^c-time channels with advantage more than n^{-c} for any $c > 0$ because for each codeword, the payload part is masked by a random string μ, and the control-information part is pseudorandom due to the property of PRC. Thus, the advantage of any PPT channel in Game_3 is close to that in Game_2 within n^{-c}.

Lemma 1. *Let $c > 0$ be any constant. For any n^c-time channel, the advantage in Game_4 is close to that in Game_3 within n^{-c}.*

Proof. It is necessary to prove that the response to the oracle queries can be replaced with the responses using the list L_Q.

Assume that there exists an n^c-time channel W for which the advantage in Game_4 is greater than that in Game_3 by n^{-c}. By fixing the random coins of W, there is a deterministic channel W_0 that achieves the same advantages as W.

Since W_0 is deterministic, we can prepare a list L_Q for answering the oracle queries by W_0. The list L_Q is constructed as follows. Let m_i be the i-th query to the encoding oracle, and y_i the i-th query to the decoding one. For each m_i, choose $c_i \in \{0,1\}^n$ uniformly at random, and add (m_i, c_i) to L_Q. The pair (m^*, c^*) of the challenge message and ciphertext is also added to L_Q. Then, for each y_i, we sample the number $\ell_i \in \mathbb{N}$, which represents the number of valid codewords within a distance $\lceil pn \rceil$ from y_i. The number ℓ_i is chosen according to the distribution D such that $\Pr[D = j] = p_j$ for $j \in [2^{Rn}]$, where p_j is the probability that a fixed Hamming ball of radius pn contains j codewords when all codewords are chosen uniformly at random. For chosen ℓ_i, if the number of codewords c_j in L_Q satisfying $\Delta(y_i, c_j) \leq \lceil pn \rceil$ is less than ℓ_i, add pairs (m_j, c_j) of random message and codeword to L_Q so that $\ell_i = |\{(m_j, c_j) \in L_Q | \Delta(y_i, c_j) \leq \lceil pn \rceil\}|$. Since each ℓ_j is bounded above by a polynomial in n, the size of L_Q is also bounded by a polynomial in n.

Note that, since it is necessary to generate MAC tags to generate valid codewords, W_0 cannot generate valid codewords by himself. Since each message has the unique codeword, W_0 cannot generate a valid codeword c' of a message m from the codeword $c(\neq c')$ of m obtained by querying to the encoding oracle. Thus, all valid codewords appeared in the game are included in the list L_Q.

Since every response to the encoding query looks random for W_0, the encoding oracle can be simulated by using L_Q. For each decoding query y_i, the response by using L_Q is equivalent to that of the decoding oracle of random codes. Therefore, both the encoding and the decoding oracles can be simulated successfully.

Next, we show that the channel cannot generate query y for which there is some $(m, c) \in L_Q$ satisfying $\Delta(y, c) \leq \lceil pn \rceil$, but the decoding algorithm outputs a list in which m is not included. It means that, on input y, the decoder fails to recover m. We show that the decoding algorithm can recover m for such y with high probability.

For $(m, c) \in L_Q$, let $e = y \oplus c$. For an error vector $e \in \{0,1\}^n$ and a set $V \subseteq [n]$, which specifies the positions of the control blocks, we say that V is *good* for e if there are at least $\varepsilon\kappa/2$ control blocks in which the fraction of errors is at most $p + \varepsilon$.

Let π and V be the permutation and the set that are generated in encoding m to c. Then, V is independent of e. This is because the payload part of c is pseudorandom by the random mask μ, and the control-information part of c is also pseudorandom by the property of PRC. Thus, the information on V is not revealed from c, and hence V is independent of e. Since V is chosen uniformly at random independently from e, V is good for e except with negligible probability. The analysis can be done in a similar way to the proof of [6, Lemma 7.11].

When V is good for e, due to the list-decoding property of PRC, the decoding algorithm can generate a list L_1 that contains correct symbols $f(\alpha_i)$ for at least $\varepsilon\kappa/2$ control blocks. Since the PRC decoding outputs a list at most L, the size of L_1 is at most ℓL. On input L_1, the list decoding of RS outputs a list L_2 of size $O(L/\varepsilon)$ that contains the correct control information $s = (s_\pi, s_\mu, s_V)$, which is equal to the value of PRF $F^1_{s_1}$ on input m. Given the correct value s, the correct values of π, μ, V can be recovered. By the same reason as for V, π is chosen uniformly at random independently from e. Thus, it follows from the p-error correctability of REC that the decoder of REC outputs the correct m with probability at least $1 - \mathsf{negl}(n)$.

We have proved that for any query y to the decoding oracle, (1) if there is no $(m, c) \in L_Q$ satisfying $\Delta(y, c) \leq \lceil pn \rceil$, the decoder output $\perp$; (2) if there is $(m, c) \in L_Q$ satisfying $\Delta(y, c) \leq \lceil pn \rceil$, the decoder outputs a list containing m. Therefore, the response of the decoding oracle in Game_2 can be replaced with the response using L_Q as in Game_3. The statement follows. □

Next, we show that for any PPT channel W that has the advantage ε in Game_4, there is a PPT channel W' that has the same advantage in Game_5. It is sufficient to show that W' can simulate the encoding and decoding oracles for W in Game_4. As described in the proof of Lemma 1, by preparing the list L_Q, W' can simulate the encoding and the decoding oracles for W in Game_4.

Finally, we argue that the advantage of any n^c-time channel W in Game_5 is at most n^{-c}. Note that the only valid codeword W obtains in Game_5 is the challenge codeword c^* of the challenge message m^*. As discussed in the proof of Lemma 1, it is difficult for W to generate y for which $\Delta(y, c^*) \leq \lceil pn \rceil$ and the decoder outputs $\perp$ or a list in which m^* is not included. Therefore, the advantage of any n^c-time channel in Game_5 is at most n^{-c}.

We have proved that, for any n^c-time channel, the advantages in Game_i are close each other within n^{-c} for $i \in \{0, 1, 2, 3, 4, 5\}$, and the advantage in Game_5 is at most n^{-c}. By taking a constant c sufficiently large enough, we can conclude that, for any n^c-time channel, the advantage in Game_0 is at most n^{-c}, which implies that the coding scheme is (p, n^c, n^{-c})-CCA secure. □

Acknowledgment

This research was supported in part by JSPS/MEXT Grant-in-Aid for Scientific Research Numbers 15H00851 and 16H01705.

References

1. E. Arikan. Channel polarization: a method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. *IEEE Trans. Information Theory*, 55(7):3051–3073, 2009.
2. G. D. Forney. *Concatenated Codes.* MIT research monograph, no. 37. MIT Press, Cambridge, 1966.

3. O. Goldreich, S. Goldwasser, and S. Micali. On the cryptographic applications of random functions. In G. R. Blakley and D. Chaum, editors, *Advances in Cryptology, Proceedings of CRYPTO '84, Santa Barbara, California, USA, August 19-22, 1984, Proceedings*, volume 196 of *Lecture Notes in Computer Science*, pages 276–288. Springer, 1984.
4. O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986.
5. V. Guruswami and A. Smith. Codes for computationally simple channels: Explicit constructions with optimal rate. In *FOCS*, pages 723–732. IEEE Computer Society, 2010.
6. V. Guruswami and A. Smith. Optimal-rate code constructions for computationally simple channels. *CoRR*, abs/1004.4017, 2013. This is an extended version of [5].
7. J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.
8. M. Langberg. Private codes or succinct random codes that are (almost) perfect. In *45th Symposium on Foundations of Computer Science (FOCS 2004), 17-19 October 2004, Rome, Italy, Proceedings*, pages 325–334. IEEE Computer Society, 2004.
9. R. J. Lipton. A new approach to information theory. In P. Enjalbert, E. W. Mayr, and K. W. Wagner, editors, *STACS*, volume 775 of *Lecture Notes in Computer Science*, pages 699–708. Springer, 1994.
10. S. Micali, C. Peikert, M. Sudan, and D. A. Wilson. Optimal error correction against computationally bounded noise. In J. Kilian, editor, *Theory of Cryptography, Second Theory of Cryptography Conference, TCC 2005, Cambridge, MA, USA, February 10-12, 2005, Proceedings*, volume 3378 of *Lecture Notes in Computer Science*, pages 1–16. Springer, 2005.
11. S. Micali, C. Peikert, M. Sudan, and D. A. Wilson. Optimal error correction for computationally bounded noise. *IEEE Transactions on Information Theory*, 56(11):5673–5680, 2010.
12. M. Sudan. Decoding of reed solomon codes beyond the error-correction bound. *J. Complexity*, 13(1):180–193, 1997.