PAPER

# Secure Regenerating Codes Using Linear Regenerating Codes and the All-or-Nothing Transform

**Hidenori KUWAKADO**[†a)], ***Senior Member and*** **Masazumi KURIHARA**[††b)], ***Member***

**SUMMARY** This paper proposes secure regenerating codes that are composed of non-secure regenerating codes and a new all-or-nothing transform. Unlike the previous analysis of secure regenerating codes, the security of the proposed codes is analyzed in the sense of the indistinguishability. The advantage of the proposed codes is that the overhead caused by the security against eavesdropping is much less than that of previous secure regenerating codes. The security of the proposed codes against eavesdropping mainly depends on the new all-or-nothing transform.

***key words:*** *regenerating code, secure regenerating code, all-or-nothing transform, indistinguishability*

## 1. Introduction

Cloud computing is supported with large-scale distributed storage systems. Since component failures in storage systems often happen, data are usually stored in redundant ways to ensure their availability. *Regenerating codes* [1] allow us to encode a message to *n shares* in such a way that the following properties are satisfied.

Reconstruction: The message can be reconstructed from any $k$ shares.

Regeneration: Any share can be regenerated from any $d$ *pieces* that are computed from shares.

It seems difficult to achieve this property by erasure encoding such as Reed-Solomon codes. Since the seminal paper of Dimakis *et al.* [1], many regenerating codes have been proposed, but they have imposed various restrictions in terms of parameters and practicality. Rashmi *et al.* [2] have shown the generic construction of regenerating codes, called a *product-matrix framework*. There are three types of regenerating methods: exact repair, functional repair, and hybrid repair. The regenerating method of the product-matrix framework is exact repair. While the product-matrix framework often requires a large finite field, their codes can be constructed over the binary finite field. Hence, their codes are useful in practice.

Regenerating codes were originally for improving the availability of nodes and are not for achieving the confidentiality. When the network to which nodes are connected is public, there is a possibility of eavesdropping. Pawar *et al.* [3] have studied regenerating codes for achieving the confidentiality, called *secure regenerating codes*. Secure regenerating codes are better than secret sharing schemes on distributed storage systems because the latter cannot regenerate a share without the reconstruction of a message. Regenerating codes based on the product-matrix framework can be modified to secure regenerating codes that achieve the upper bound of secrecy capacity [4]–[7]. Another way to construct secure regenerating codes is to employ an appropriate preprocessing before encoding a message with non-secure regenerating codes [8]. This code does not achieve the upper bound of secrecy capacity.

The above secure regenerating codes are information-theoretically secure against eavesdropping. However, the amount of stored data with secure regenerating codes is much less than that with non-secure regenerating codes even if the upper bound of secrecy capacity is achieved. To solve this problem, we relax the definition of secure regenerating codes using the concept of indistinguishability. An adversary (including an eavesdropper) in this paper is assumed to have unlimited computational power except for making access to oracles. We next propose a new scheme for constructing secure regenerating codes. The proposed scheme consists of a non-secure linear regenerating code and an all-or-nothing transform. The amount of stored data with the proposed secure regenerating code is almost the same as that with the underlying non-secure regenerating code.

The all-or-nothing transform is one such that if a part of the output is missing, then no information about the input is given. Although all-or-nothing transforms were designed to improve the security against brute-force attacks of a secret key [9], the all-or-nothing transform used in this paper is newly designed for constructing secure regenerating codes.

This paper is organized as follows: Section 2 briefly summarizes the concept of regenerating codes, the definition of secure regenerating codes, and the all-or-nothing transform. Section 3 proposes how to construct a secure regenerating code from an all-or-nothing transform and a non-secure regenerating code. Section 3 also shows an example of the construction that does not yield the secure regenerating code. Since this is primarily caused by the definition of an all-or-nothing transform, we need a new definition suitable for our purpose. Section 4 relaxes the definition of an all-or-nothing transform and shows a new all-or-nothing

Manuscript received May 26, 2016.
Manuscript revised September 26, 2016.
Manuscript publicized December 6, 2016.

[†]The author is with the Faculty of Informatics, Kansai University, Takatsuki-shi, 569–1095 Japan.
[††]The author is with the Graduate School of Informatics and Engineering, The University of Electro-Communications, Chofu-shi, 182–8585 Japan.
a) E-mail: kuwakado@kansai-u.ac.jp
b) E-mail: kurihara@uec.ac.jp

DOI: 10.1587/transinf.2016EDP7220

transform satisfying the relaxed definition. Its proof is given in Sect. 6. Section 5 proposes a new definition of secure regenerating codes and shows a new secure regenerating code satisfying the new definition. The comparison of new secure regenerating codes with previous secure regenerating codes is described. Section 7 concludes this paper.

## 2. Preliminaries

This section summarizes notations and definitions on regenerating codes, secure regenerating codes, and all-or-nothing transforms. Furthermore, their examples are given.

### 2.1 Regenerating Codes

Consider a distributed storage system with $n$ nodes for storing a message $(z_1, \ldots, z_B) \in \mathbb{F}_p^B$ where $\mathbb{F}_p$ denotes a finite field with $p$ elements. Each node stores $\alpha$ symbols called a *share*, denoted by

$$c^{(i)} = (c_1^{(i)}, \ldots, c_\alpha^{(i)}) \in \mathbb{F}_p^\alpha \tag{1}$$

where $i$ denotes a node index. A data collector can reconstruct the message from any $k$ shares. When a node $f$ fails, a replacement node connects to any $d$ remaining nodes called *helper nodes*, and downloads $\beta$ symbols called a *piece* from each helper node, denoted by

$$d_f^{(i)} = (d_{f,1}^{(i)}, \ldots, d_{f,\beta}^{(i)}) \in \mathbb{F}_p^\beta. \tag{2}$$

Note that the helper node $i$ can compute $d_f^{(i)}$ from $c^{(i)}$ and the replacement node can regenerate $c^{(f)}$ from an ordered set of $d$ pieces

$$v_f^{(i_1,\ldots,i_d)} = (d_f^{(i_1)}, \ldots, d_f^{(i_d)}), \tag{3}$$

which is called a *piece vector* for node $f$. The size $d\beta$ of a piece vector is called *repair bandwidth*.

An $[n, k, d, \alpha, \beta, B]$ regenerating code is a code such that $k$ and $d$ are minimum values under which the message reconstruction and the share regeneration can be always guaranteed. When parameters $\alpha, \beta, B$ are not focused, they are sometimes omitted. That is, it is sometimes called an $[n, k, d]$ regenerating code. We call the regenerating code such that

$$\alpha = \frac{2dB}{k(2d-k+1)}, \quad \beta = \frac{2B}{k(2d-k+1)} \tag{4}$$

a *minimum bandwidth regeneration (MBR) code* and call the regenerating code such that

$$\alpha = \frac{B}{k}, \quad \beta = \frac{B}{k(d-k+1)} \tag{5}$$

a *minimum storage regeneration (MSR) code*.

After several constructions of MBR/MSR codes for limited parameters were studied, Rashmi *et al.* [2] have shown the *product-matrix framework* to construct MBR/MSR codes for wide range parameters. We call their codes *PM-MBR/MSR* codes. The more general construction of PM-MSR codes has been shown by Lin and Chung [10].

### 2.2 Special Class of the MBR Code

A *repair-by-transfer (RBT) MBR code* is the MBR code such that the regeneration is accomplished only with transfer of pieces [11]. For this remarkable feature, a piece vector for failed node $f$ is equal to a share of failed node $f$. That is,

$$v_f^{(i_1,\ldots,i_d)} = c^{(f)}, \tag{6}$$

which is equivalent to

$$(d_{f,1}^{(i_1)}, \ldots, d_{f,\beta}^{(i_1)}, \ldots, d_{f,1}^{(i_d)}, \ldots, d_{f,\beta}^{(i_d)}) = (c_1^{(f)}, \ldots, c_\alpha^{(f)})$$

because of Eqs. (1)–(3). From the equation above, we see $d\beta = \alpha$, which is the condition of MBR codes (Eq. (4)).

For example, an $[n, k, n-1, n-1, 1, B]$ RBT-MBR code is constructed in the following manner. A message is denoted by $(z_1, \ldots, z_B) \in \mathbb{F}_p^B$. Encode $B$ message symbols using an $[\binom{n}{2}, B]$-MDS code[†] where $\binom{n}{2}$ and $B$ denote a codeword length and the number of information symbols, respectively. Let $\{c_i | i = 1, \ldots, \binom{n}{2}\}$ be code symbols. Assign each code symbol to nodes in such a way that each code symbol is stored in two nodes. Accordingly, each node has $n-1$ code symbols that are a share of the node. When a node fails, each helper node just transmits the code symbol that was shared with the failed node.

We show a $[4, 2, 3, 3, 1, 5]$ RBT-MBR code for $(z_1, \ldots, z_5) \in \mathbb{F}_p^5$, which will be reused in Sect. 3. Using a single parity check code of length 6, we encode the message $(z_1, \ldots, z_5)$ to a codeword $(c_1, \ldots, c_5, c_6)$ as

$$c_i = \begin{cases} z_i & i = 1, \ldots, 5 \\ z_1 + z_2 + \cdots + z_5 \text{ over } \mathbb{F}_p & i = 6. \end{cases} \tag{7}$$

This single parity check code is the $[6, 5]$-MDS code. Then, node $i$ stores a share $c^{(i)}$ that consists of three code symbols as follows:

$$c^{(1)} = (c_1, c_2, c_3), \quad c^{(2)} = (c_1, c_4, c_5),$$
$$c^{(3)} = (c_2, c_5, c_6), \quad c^{(4)} = (c_3, c_4, c_6).$$

We see that the message can be reconstructed from any two shares and any share can be regenerated by three nodes. For example, if node 4 fails, then a replacement node can regenerate share $c^{(4)}$ by receiving $c_3, c_4, c_6$ from nodes 1, 2, 3, respectively. That is, the piece vector $v_4^{(1,2,3)}$ and pieces $d_4^{(1)}$, $d_4^{(2)}$, $d_4^{(3)}$ are given as

$$\begin{aligned} v_4^{(1,2,3)} &= (d_4^{(1)}, d_4^{(2)}, d_4^{(3)}) \\ d_4^{(1)} &= (d_{4,1}^{(1)}) = (c_3) \\ d_4^{(2)} &= (d_{4,1}^{(2)}) = (c_4) \\ d_4^{(3)} &= (d_{4,1}^{(3)}) = (c_6). \end{aligned} \tag{8}$$

[†]Maximum distance separable code.

We can confirm that Eq. (6) holds, that is, $v_4^{(1,2,3)} = c^{(4)}$. Since $\beta = 1$, the piece is indeed a scalar value rather than a vector.

### 2.3 Linear Regenerating Codes

PM-MBR/MSR codes and RBT-MBR codes belong to a class of *linear regenerating codes*, which is defined as a set of regenerating codes satisfying the following two conditions.

1. Linear reconstruction: Code symbols stored in each node are linear combinations of the $B$ message symbols over $\mathbb{F}_p$.
2. Linear regeneration: Symbols in the piece of a helper node are linear combinations of $\alpha$ symbols of its share over $\mathbb{F}_p$.

We here focus on the linear reconstruction because the linear regeneration is of little relevance to later discussion. We can rephrase the linear reconstruction (the first condition) as follows: There exists a matrix $\mathbf{M}_{\mathrm{e}}$ that is independent of a message $(z_1, \ldots, z_B) \in \mathbb{F}_p^B$ such that

$$\begin{pmatrix} c_1^{(1)} \\ \vdots \\ c_\alpha^{(1)} \\ \vdots \\ c_1^{(n)} \\ \vdots \\ c_\alpha^{(n)} \end{pmatrix} = \mathbf{M}_{\mathrm{e}} \begin{pmatrix} z_1 \\ \vdots \\ z_B \end{pmatrix} \text{ over } \mathbb{F}_p,$$

where $(c_1^{(i)}, \ldots, c_\alpha^{(i)})$ denotes a share $c^{(i)}$ of node $i$. In this paper, we interchangeably use a row vector and a column vector. Since the interchange is very obvious, it does not create any confusion. For example, $\mathbf{M}_{\mathrm{e}}$ of the $[4, 2, 3, 3, 1, 5]$ RBT-MBR code described in Sect. 2.2 is given as

$$\mathbf{M}_{\mathrm{e}} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

Similarly, we can rephrase the reconstruction as follows: There exists a way for solving the following system of equations in terms of a message $(z_1, \ldots, z_B)$.

$$\begin{pmatrix} c_1^{(i_1)} \\ \vdots \\ c_\alpha^{(i_1)} \\ \vdots \\ c_1^{(i_k)} \\ \vdots \\ c_\alpha^{(i_k)} \end{pmatrix} = \mathbf{M}_{\mathrm{rc}}^{(i_1,\ldots,i_k)} \begin{pmatrix} z_1 \\ \vdots \\ z_B \end{pmatrix} \text{ over } \mathbb{F}_p \tag{9}$$

where $\mathbf{M}_{\mathrm{rc}}^{(i_1,\ldots,i_k)}$ denotes a matrix that consists of rows of $\mathbf{M}_{\mathrm{e}}$ corresponding to $c^{(i_1)}, \ldots, c^{(i_k)}$. Since $\mathbf{M}_{\mathrm{rc}}^{(i_1,\ldots,i_k)}$ is usually a non-square matrix, $\mathbf{M}_{\mathrm{e}}$ is artfully designed to find the message. For example, consider the RBT-MBR code described in Sect. 2.2. When the data collector has access to node 1 and node 4, $\mathbf{M}_{\mathrm{rc}}^{(1,4)}$ is given by

$$\mathbf{M}_{\mathrm{rc}}^{(1,4)} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

Although the matrix above is not non-singular, it is easy to see that the message $(z_1, \ldots, z_5)$ can be uniquely determined for a given $(c_1^{(1)}, c_2^{(1)}, c_3^{(1)}, c_1^{(4)}, c_2^{(4)}, c_3^{(4)})$.

Let $v$ be an integer less than $k$ (i.e., $v < k$). From the definition of regenerating codes, there is no way for determining $(z_1, \ldots, z_B)$ uniquely to satisfy the following system of equations.

$$\begin{pmatrix} c_1^{(i_1)} \\ \vdots \\ c_\alpha^{(i_1)} \\ \vdots \\ c_1^{(i_v)} \\ \vdots \\ c_\alpha^{(i_v)} \end{pmatrix} = \mathbf{M}_{\mathrm{rc}}^{(i_1,\ldots,i_v)} \begin{pmatrix} z_1 \\ \vdots \\ z_B \end{pmatrix} \text{ over } \mathbb{F}_p$$

where $\mathbf{M}_{\mathrm{rc}}^{(i_1,\ldots,i_v)}$ is a matrix that consists of rows of $\mathbf{M}_{\mathrm{e}}$ corresponding to $c^{(i_1)}, \ldots, c^{(i_v)}$. For example, consider the RBT-MBR code described in Sect. 2.2. When a data collector only has access to node 3, $\mathbf{M}_{\mathrm{rc}}^{(3)}$ is given by

$$\mathbf{M}_{\mathrm{rc}}^{(3)} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}. \tag{10}$$

Although message symbols $z_2$, $z_5$ can be found, the others $z_1$, $z_3$, $z_4$ cannot be determined uniquely, that is, the data collector only knows the value of $z_1 + z_3 + z_4$. Thus, when one share is missing, one message symbol is not always lost and a linearly-combined value of lost message symbols may be given.

### 2.4 Secure Regenerating Codes

**Definition 1** ([4]). *Let $\ell$ be an integer less than $k$ and let $\tau$ and $\mu$ denote non-negative integers such that $\ell = \tau + \mu$. Suppose that an eavesdropper gains access to shares in $\tau$ nodes, and the data downloaded during repair (i.e., piece vectors) in other $\mu$ nodes. An $\{\ell, \mu\}$ secure distributed storage system is one in which such an eavesdropper obtains no information about the message.*

We call an $[n, k, d]$ regenerating code for achieving the $\{\ell, \mu\}$ secure distributed storage system an $[n, k, d, \alpha, \beta, B_s, \ell, \mu]$ *secure regenerating code*. The case of $\ell = \mu$ was studied by Pawar *et al.* [3] before Definition 1 was proposed.

Secure regenerating codes based on PM-MBR/MSR codes, called *PM-SR* codes, have been proposed in articles [4]–[7]. The idea of PM-SR codes is to replace $R$ message symbols in $B$ message symbols of an $[n, k, d, \alpha, \beta, B]$ PM-MBR/MSR code with $R$ random symbols. The following examples of PM-SR codes will be used for a comparison with the proposed schemes in Sect. 5.2.1 and Sect. 5.2.2.

(1) PM-SR Code Using a PM-MBR Code

Given an $[n, k, d, d, 1, k(2d-k+1)/2]$ PM-MBR code, replacing $(k-1)(2d-k+2)/2$ appropriate message symbols with random symbols yields an $[n, k, d, d, 1, d-k+1, k-1, \mu]$ PM-SR code where $\mu \leq k-1$ [5]. For example, replacing 15 appropriate message symbols of the $[n, 4, 6, 6, 1, 18]$ PM-MBR code with 15 random symbols yields the $[n, 4, 6, 6, 1, 3, 3, \mu]$ PM-SR code where $\mu \leq 3$.

(2) PM-SR Code Using a PM-MSR Code

Given an $[n, k, 2(k-1), k-1, 1, k(k-1)]$ PM-MSR code, replacing $2(k-1)$ appropriate message symbols with random symbols yields an $[n, k, 2(k-1), k-1, 1, (k-1)(k-2), \ell, \mu]$ PM-SR code where $\ell + \mu \leq 2$ [7]. For example, replacing 8 appropriate message symbols of the $[n, 5, 8, 4, 1, 20]$ PM-MSR code with 8 random symbols yields the $[n, 5, 8, 4, 1, 12, \ell, \mu]$ PM-SR code where $\ell + \mu \leq 2$.

Both of the PM-SR codes achieve the upper bound of the secrecy capacity defined by Pawar *et al.* [3] That is, we cannot make the number of message symbols larger as long as we obey Definition 1. The objective of this paper is to increase the number of message symbols.

### 2.5 All-or-Nothing Transform

An all-or-nothing transform $\Pi$ is defined as a pair of two algorithms $(\mathcal{E}, \mathcal{D})$ satisfying the following conditions [9].

1. $\mathcal{E}$ is a probabilistic algorithm that takes a message $(m_1, \ldots, m_B) \in \{0,1\}^{lB}$ to produce a pseudo-message $(z_1, \ldots, z_{B'}) \in \{0,1\}^{lB'}$ where $l$ is a bit length of a symbol $z_i$. Note that $l$ and $\ell$ are different symbols.
2. $\mathcal{D}$ is a deterministic algorithm that takes a pseudo-message $(z_1, \ldots, z_{B'})$ to produce the corresponding message $(m_1, \ldots, m_B)$ or a special symbol $\perp$ where $\perp$ indicates that the pseudo-message is invalid.
3. It is infeasible to compute any function of any message symbol $m_i$ if any one of the pseudo-message symbols $z_i$ is unknown.

The all-or-nothing transform $\Pi$ does not require any secret information. That is, everyone can compute the pseudo-message from a message and also can obtain the message (or $\perp$) from a pseudo-message. In this sense, the all-or-nothing transform is not an encryption.

Since the last condition is informal, Desai [12] has formally defined the infeasibility using an experiment. The experiment consists of a find stage and a guess stage. During the find stage, it comes up with a message and some auxiliary information. A challenge is either its pseudo-message or a random string. In the guess stage, an adversary is allowed to obtain all but one of the challenge symbols and it guesses whether the challenge is the pseudo-message or the random string.

Before introducing Desai's definition, we describe notations. For a probabilistic algorithm $H$, $h \leftarrow H$ denotes that the output of $H$ is assigned to $h$. For a set $S$, $s \twoheadleftarrow S$ denotes that an element is chosen from $S$ according to the uniform distribution on $S$ and the element is assigned to $s$.

**Definition 2** ([12]). *Let $\Pi = (\mathcal{E}, \mathcal{D})$. For a probabilistic algorithm (an adversary) $A$ and $b \in \{0,1\}$, define an experiment $\mathrm{Exp}^{\mathrm{aon}}_{\Pi}(A, b)$ as follows.*

1. find *stage:* $((m_1, \ldots, m_B), s) \leftarrow A$ *where $s$ is auxiliary information. $A$ can transmit information such as hints with $s$ to a* guess *stage.*
2. *If $b = 0$, then $(z_1, \ldots, z_{B'}) \leftarrow \mathcal{E}((m_1, \ldots, m_B))$. Otherwise $(z_1, \ldots, z_{B'}) \twoheadleftarrow \{0,1\}^{lB'}$.*
3. guess *stage:* $b_A \leftarrow A^{\mathcal{O}}(s)$ *where*
   - *$\mathcal{O}$ denotes an oracle that takes $B'-1$ indexes $j \in \{1, \ldots, B'\}$ returns $z_j$'s.*
   - *$s$ denotes the auxiliary information produced in the* find *stage.*

   *Unlike the* find *stage, the adversary $A$ of this stage only outputs 0 or 1.*
4. *The value of $\mathrm{Exp}^{\mathrm{aon}}_{\Pi}(A, b)$ is $b_A$.*

*The advantage of $A$ is defined as*

$$\begin{aligned}&\mathrm{Adv}^{\mathrm{aon}}_{\Pi}(A)\\ &= \left|\Pr\left[\mathrm{Exp}^{\mathrm{aon}}_{\Pi}(A,1) = 1\right] - \Pr\left[\mathrm{Exp}^{\mathrm{aon}}_{\Pi}(A,0) = 1\right]\right|\\ &= \left|2\Pr\left[\mathrm{Exp}^{\mathrm{aon}}_{\Pi}(A,b) = b\right] - 1\right|,\end{aligned}$$

*and the advantage of $\Pi$ is defined as*

$$\mathrm{Adv}^{\mathrm{aon}}_{\Pi}(q) = \max_{A} \mathrm{Adv}^{\mathrm{aon}}_{\Pi}(A), \tag{11}$$

*where $q$ is the number of queries to underlying oracles that are used by $\mathcal{E}$ and $\mathcal{D}$ if they exist. If $\mathrm{Adv}^{\mathrm{aon}}_{\Pi}(q)$ is not larger than some criterion, then $\Pi$ is an all-or-nothing transform in the sense of this definition.*

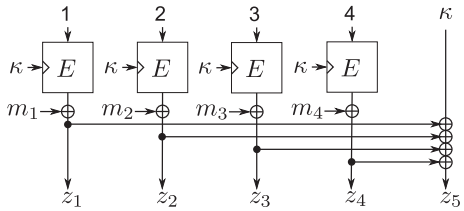


**Fig. 1** Desai's all-or-nothing transform ($B = 4$).

Desai [12] has proposed the all-or-nothing transform $\Pi_{\mathrm{D}}$ in the sense of Definition 2. The algorithm $\mathcal{E}$ of $\Pi_{\mathrm{D}}$ is as follows (Fig. 1): Let $E$ be an ideal cipher, that is, a cipher that is uniformly chosen from a set of all blockciphers $\{0,1\}^l \times \{0,1\}^l \to \{0,1\}^l$ at random. Let $\kappa \xleftarrow{} \{0,1\}^l$. For a given message $(m_1, \ldots, m_B) \in \{0,1\}^{lB}$, compute $z_i$ as

$$z_i = \begin{cases} m_i \oplus E(\kappa, i) & \text{if } i = 1, \ldots, B \\ \kappa \oplus z_1 \oplus z_2 \oplus \ldots \oplus z_B & \text{if } i = B+1, \end{cases} \tag{12}$$

where $\oplus$ denotes the bitwise XOR operator and $i$ denotes the $l$-bit binary string for representing $i$. We use the same symbol for representing a value itself and for representing its binary string with appropriate length. The pseudo-message is given by $(z_1, \ldots, z_{B+1})$. The algorithm $\mathcal{D}$ of $\Pi_{\mathrm{D}}$ is omitted. Remark: The discussion in this paper is based on the ideal-cipher model. A computational model such as a pseudo-random permutation is generally preferable to the ideal-cipher model. Although the idealization may not be an accurate reflection of reality, we can at least derive some measure of the security from a proof within the ideal-cipher model.

## 3. Counter Example

Consider the following scheme (Fig. 2). First, compute the pseudo-message from a message using an all-or-nothing transform. Next, produce shares from the pseudo-message using an $[n, k, d]$ regenerating code. Then, owing to the reconstruction property of the regenerating code, if $k$ shares are not collected, then the pseudo-message cannot be reconstructed completely. Since a part of the pseudo-message is missing, no information about the message is given owing to the property of the all-or-nothing transform. Accordingly, one may think that combining any all-or-nothing transform and any regenerating code yields a secure regenerating code.

However, the conjecture above is incorrect. A counter example is shown below. Consider the following scheme $\Omega_{\mathrm{NG}}$ (Fig. 3). Suppose that $p = 2^l$. Let $(m_1, \ldots, m_4) \in \mathbb{F}_{2^l}^4$ be a message. We equate the bitwise XOR operation over $\{0,1\}^l$ with the addition over $\mathbb{F}_{2^l}$. Let $E$ be the encryption of an ideal cipher from $\{0,1\}^l \times \{0,1\}^l$ to $\{0,1\}^l$. The pseudo-message $(z_1, \ldots, z_5)$ is produced with Desai's all-or-nothing transform $\Pi_{\mathrm{D}}$. That is, according to Eq. (12),

$$z_i = \begin{cases} m_i + E(\kappa, i) \text{ over } \mathbb{F}_{2^l} & \text{if } i = 1, \ldots, 4 \\ \kappa + \displaystyle\sum_{i=1}^{4} z_i \text{ over } \mathbb{F}_{2^l} & \text{if } i = 5, \end{cases} \tag{13}$$

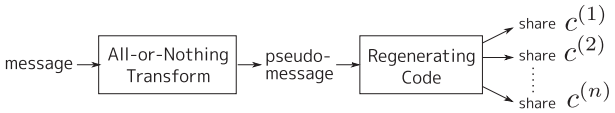


**Fig. 2** The composite scheme.

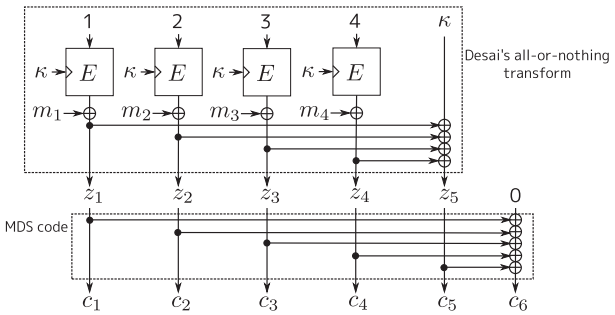


**Fig. 3** The composite scheme $\Omega_{\mathrm{NG}}$.

where $\kappa \xleftarrow{} \{0,1\}^l$ and we equate the addition over $\mathbb{F}_{2^l}$ with the bitwise XOR operation over $\{0,1\}^l$. After that, the pseudo-message is encoded by using the $[4,2,3,3,1,5]$ RBT-MBR code described in Sect. 2.2. That is, for $i = 1, \ldots, 4$, the share $c^{(i)}$ of node $i$ is given by

$$c^{(1)} = (c_1, c_2, c_3), \quad c^{(2)} = (c_1, c_4, c_5),$$
$$c^{(3)} = (c_2, c_5, c_6), \quad c^{(4)} = (c_3, c_4, c_6),$$

where $c_i$ is the code symbol of the single parity code such as

$$c_i = \begin{cases} z_i & \text{if } i = 1, \ldots, 5 \\ \displaystyle\sum_{i=1}^{5} z_i \text{ over } \mathbb{F}_{2^l} & \text{if } i = 6. \end{cases} \tag{14}$$

The above scheme $\Omega_{\mathrm{NG}}$ is the $[4,2,3,3,1,4]$ MBR code, but is not a secure regenerating code for the following attack. Suppose that an adversary only knows node 3's share $c^{(3)}$. Since $c_6$ in $c^{(3)}$ turns out to be

$$\begin{aligned} c_6 &= \sum_{i=1}^{5} z_i \text{ over } \mathbb{F}_{2^l} \quad (\because \text{Eq. (14)}) \\ &= \sum_{i=1}^{4} z_i + \left( \sum_{i=1}^{4} z_i + \kappa \right) \text{ over } \mathbb{F}_{2^l} \quad (\because \text{Eq. (13)}) \\ &= \kappa, \end{aligned}$$

the adversary can obtain the second message symbol $m_2$ by

$$m_2 = E(c_6, 2) + c_2 \text{ over } \mathbb{F}_{2^l} \quad (\because \text{Eq. (13)}).$$

Hence, the scheme $\Omega_{\mathrm{NG}}$ is not a secure regenerating code.

## 4. New All-or-Nothing Transform

### 4.1 New Definition

Definition 2 is insufficient for discussing the composite

scheme such as Fig. 2 because it assumes only the case that one pseudo-message symbol is lost. That is, Definition 2 cannot address the situation mentioned in the last paragraph of Sect. 2.3.

To solve this problem, we relax Definition 2. The relaxed definition differs in the followings.

- An adversary is given symbols to which pseudo-message symbols are linearly transformed. The linear transform is specified by the adversary, but it is not invertible.
- The relaxed definition is concerned with the indistinguishability of a pseudo-message from another pseudo-message, whereas Definition 2 is concerned with the indistinguishability of a pseudo-message from a random string. According to the relaxed definition, a pseudo-message may not look like a random string.

**Definition 3.** *Let* $\Pi = (\mathcal{E}, \mathcal{D})$*. For a probabilistic algorithm (an adversary)* $A$ *and* $b \in \{0, 1\}$*, define an experiment* $\mathrm{Exp}^{\mathrm{aon2}}_{\Pi}(A, b)$ *as follows.*

1. find *stage:* $((m_1, \ldots, m_B), (\hat{m}_1, \ldots, \hat{m}_B), s) \leftarrow A$ *where* $s$ *is auxiliary information.*
2. *If* $b = 0$*, then* $(z_1, \ldots, z_{B'}) \leftarrow \mathcal{E}((m_1, \ldots, m_B))$*. Otherwise* $(z_1, \ldots, z_{B'}) \leftarrow \mathcal{E}((\hat{m}_1, \ldots, \hat{m}_B))$*.*
3. guess *stage:* $b_A \leftarrow A^{O}(s)$ *where oracle* $O$ *takes a* $(B' - 1) \times B'$ *matrix* $\mathbf{U}$ *such as*

$$\mathbf{U} = \begin{pmatrix} u_{1,1} & u_{1,2} & \ldots & u_{1,B'} \\ u_{2,1} & u_{2,2} & \ldots & u_{2,B'} \\ \vdots & & & \\ u_{B'-2,1} & u_{B'-2,2} & \ldots & u_{B'-2,B'} \\ u_{B'-1,1} & u_{B'-1,2} & \ldots & u_{B'-1,B'} \end{pmatrix}.$$

*Notice that* $\mathbf{U}$ *is not a square matrix and* $B' - 1$ *rows in* $\mathbf{U}$ *are not always linearly independent. Oracle* $O$ *returns* $v_1, \ldots, v_{B'-1}$ *given by*

$$\begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_{B'-2} \\ v_{B'-1} \end{pmatrix} = \mathbf{U} \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_{B'-2} \\ z_{B'-1} \\ z_{B'} \end{pmatrix} \text{ over } \mathbb{F}_p. \tag{15}$$

4. *The value of* $\mathrm{Exp}^{\mathrm{aon2}}_{\Pi}(A, b)$ *is* $b_A$*.*

*The advantage of* $A$ *is defined as*

$$\begin{aligned} &\mathrm{Adv}^{\mathrm{aon2}}_{\Pi}(A) \\ &= \left|\Pr\left[\mathrm{Exp}^{\mathrm{aon2}}_{\Pi}(A, 1) = 1\right] - \Pr\left[\mathrm{Exp}^{\mathrm{aon2}}_{\Pi}(A, 0) = 1\right]\right| \\ &= \left|2\Pr\left[\mathrm{Exp}^{\mathrm{aon2}}_{\Pi}(A, b) = b\right] - 1\right|, \end{aligned} \tag{16}$$

*and the advantage of* $\Pi$ *is defined as*

$$\mathrm{Adv}^{\mathrm{aon2}}_{\Pi}(q) = \max_{A} \mathrm{Adv}^{\mathrm{aon2}}_{\Pi}(A), \tag{17}$$

*where* $q$ *is the number of queries to underlying oracles that are used by* $\mathcal{E}$ *and* $\mathcal{D}$ *if they exist. If* $\mathrm{Adv}^{\mathrm{aon2}}_{\Pi}(q)$ *is not larger than some criterion, then* $\Pi$ *is an all-or-nothing transform in the sense of this definition.*

Definition 3 can capture a wide variety of how to lose pseudo-message symbols. For example, when an adversary chooses $\mathbf{U}$ as

$$\mathbf{U} = \begin{pmatrix} 1 & 0 & \ldots & 0 & 0 \\ 1 & 1 & \ldots & 0 & 0 \\ \vdots & & & & \\ 0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & \ldots & 1 & 0 \end{pmatrix},$$

only $z_{B'}$ is unknown and others $z_1, \ldots z_{B'-1}$ are known to the adversary. This situation is similar to that of Definition 2.

### 4.2 Gap between Definitions

There exists a non-trivial gap between Definition 2 and Definition 3. Desai's all-or-nothing transform $\Pi_{\mathrm{D}}$ has been proved to be an all-or-nothing transform in the sense of Definition 2. However, we here show that $\Pi_{\mathrm{D}}$ is not an all-or-nothing transform in the sense of Definition 3. That is, $\Pi_{\mathrm{D}}$ is an example for showing the non-trivial gap between Definition 2 and Definition 3.

Consider $\Pi_{\mathrm{D}}$ for $B = 4$. According to experiment $\mathrm{Exp}^{\mathrm{aon2}}_{\Pi_{\mathrm{D}}}(A, b)$, we describe an algorithm of an adversary $A$.

1. $A$ chooses two messages $(m_1, m_2, m_3, m_4)$, $(\hat{m}_1, \hat{m}_2, \hat{m}_3, \hat{m}_4)$ such that $m_2 \neq \hat{m}_2$. Auxiliary information $s$ is unnecessary.
2. This step is not performed by $A$. The oracle $O$ computes $(z_1, \ldots, z_5)$ according to the value of $b$.
3. $A$ makes the following query to oracle $O$.

$$\mathbf{U} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{18}$$

Then, $O$ computes

$$\begin{aligned} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{pmatrix} &= \mathbf{U} \begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \\ z_5 \end{pmatrix} \text{ over } \mathbb{F}_{2^l} \\ &= \begin{pmatrix} z_2 \\ z_5 \\ z_1 + z_2 + z_3 + z_4 + z_5 \\ 0 \end{pmatrix} \text{ over } \mathbb{F}_{2^l}. \end{aligned} \tag{19}$$

After $A$ receives $v_1, v_2, v_3, v_4$ from $O$, $A$ finds $\kappa$ as

$$v_3 = \sum_{i=1}^{5} z_i \text{ over } \mathbb{F}_{2^l}$$

$$= \left(\sum_{i=1}^{4} z_i\right) + z_5 \text{ over } \mathbb{F}_{2^l}$$
$$= \left(\sum_{i=1}^{4} z_i\right) + \left(\kappa + \sum_{i=1}^{4} z_i\right) \text{ over } \mathbb{F}_{2^l} (\because \text{Eq. (13)})$$
$$= \kappa.$$

$A$ knows $v_1 = z_2$ because of Eq. (19). If

$$v_1 = m_2 + E(\kappa, 2) \text{ over } \mathbb{F}_{2^l} \tag{20}$$

holds, then $A$ assigns 0 to $b_A$. If

$$v_1 = \hat{m}_2 + E(\kappa, 2) \text{ over } \mathbb{F}_{2^l} \tag{21}$$

holds, then $A$ assigns 1 to $b_A$. Since $m_2 \neq \hat{m}_2$ and Eq. (13), Eq. (20) or Eq. (21) not both holds.

4. The value of $\mathrm{Exp}^{\mathrm{aon2}}_{\Pi_{\mathrm{D}}}(A, b)$ is $b_A$.

We evaluate the advantage of $A$. If $b = 0$ in the step 2, then Eq. (20) always holds, resulting in $\Pr\left[\mathrm{Exp}^{\mathrm{aon2}}_{\Pi_{\mathrm{D}}}(A, 0) = 1\right] = 0$. If $b = 1$ in the step 2, then Eq. (21) always holds, resulting in $\Pr\left[\mathrm{Exp}^{\mathrm{aon2}}_{\Pi_{\mathrm{D}}}(A, 1) = 1\right] = 1$. Hence, the advantage of $A$ is given by

$$\mathrm{Adv}^{\mathrm{aon2}}_{\Pi}(A)$$
$$= \left|\Pr\left[\mathrm{Exp}^{\mathrm{aon2}}_{\Pi_{\mathrm{D}}}(A, 1) = 1\right] - \Pr\left[\mathrm{Exp}^{\mathrm{aon2}}_{\Pi_{\mathrm{D}}}(A, 0) = 1\right]\right|$$
$$= 1.$$

Thus, the adversary can always guess the value of $b$ correctly. In fact, the algorithm above is equivalent to that of Sect. 3. The matrix $\mathbf{U}$ of Eq. (18) is essentially the same as the matrix $\mathbf{M}^{(3)}_{\mathrm{rc}}$ of Eq. (10).

### 4.3 New Algorithm

We here describe a new all-or-nothing transform $\Pi_{\mathrm{new}} = (\mathcal{E}, \mathcal{D})$. Suppose that $E$ is the encryption of an ideal cipher $\{0,1\}^{\zeta} \times \{0,1\}^l \to \{0,1\}^l$ where $\zeta \geq l + \log_2(2B)$. The algorithm of $\mathcal{E}$ is as follows (Fig. 4): Let $(m_1, \ldots, m_B)$ be a message where $m_i \in \{0,1\}^l$. Choose a pseudo-key $\kappa$ from $\{0,1\}^l$ uniformly at random. For $i = 1, \ldots, B$, compute a pseudo-message symbol $z_i$ as

$$z_i = E(\kappa \parallel i, m_i), \tag{22}$$

where $\parallel$ denotes the concatenation operator of strings. Next,

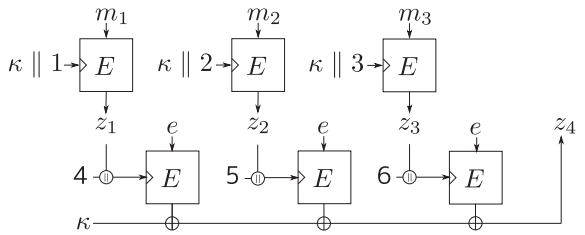


**Fig. 4** The proposed all-or-nothing transform ($B = 3$).

for $i = 1, \ldots, B$, compute $w_i$ as

$$w_i = E(z_i \parallel (B + i), e), \tag{23}$$

where $e$ is a public constant value (say, all zero bits). Compute the last pseudo-message symbol $z_{B+1}$ as

$$z_{B+1} = \kappa \oplus w_1 \oplus \ldots \oplus w_B \tag{24}$$
$$= \kappa + \sum_{i=1}^{B} w_i \text{ over } \mathbb{F}_{2^l}.$$

Finally, output $z = (z_1, \ldots, z_{B+1})$ as a pseudo-message. The algorithm of $\mathcal{D}$ is omitted.

Lemma 1 implies that $\Pi_{\mathrm{new}}$ is the all-or-nothing transform in each of new definitions if the number of queries to the ideal cipher is much less than $2^l$. Their proofs are given in Sect. 6.

**Lemma 1.** *The advantage of* $\Pi_{\mathrm{new}}$ *in the sense of Definition 3 is given by*

$$\mathrm{Adv}^{\mathrm{aon2}}_{\Pi_{\mathrm{new}}}(q) \leq \frac{2(2B+3)q}{2^l - q}, \tag{25}$$

*where $q$ is the number of queries to the ideal cipher.*

## 5. New Secure Regenerating Codes

### 5.1 New Definition of Secure Regenerating Codes

Definition 1 is the information-theoretical definition of secure regenerating codes. We relax the definition of secure regenerating codes in a manner similar to Definition 3.

**Definition 4.** *Let $\Omega = (\mathcal{P}, C, \mathcal{R})$ be an $[n, k, d, \alpha, \beta, B]$ regenerating code where $\mathcal{P}$, $\mathcal{R}$, and $C$ denote an encoding, a reconstructing, and a regenerating algorithm, respectively. An experiment for an adversary $A$, denoted by $\mathrm{Exp}^{\mathrm{sr}}_{\Omega}(A, b)$, is defined as follows.*

1. find *stage:* $((m_1, \ldots, m_B), (\hat{m}_1, \ldots, \hat{m}_B), s) \leftarrow A$ *where message symbols $m_i$ are elements in $\mathbb{F}_p$ and $s$ denotes auxiliary information. If $\Omega$ uses oracles $O_p$, then $A$ can make queries to oracles.*
2. *If $b = 0$, then $C \leftarrow \mathcal{P}((m_1, \ldots, m_B))$, otherwise $C \leftarrow \mathcal{P}((\hat{m}_1, \ldots, \hat{m}_B))$.*
3. guess *stage:* $b_A \leftarrow A^{O}(s)$ *where oracle $O$ accepts queries about $\tau$ shares $c^{(i_1)}, \ldots, c^{(i_\tau)}$ and $\mu$ piece vectors*

$$v_{f_1}^{(i_{f_1,1},\ldots,i_{f_1,d})}, \ldots, v_{f_\mu}^{(i_{f_\mu,1},\ldots,i_{f_\mu,d})}$$

*and returns it. If $\Omega$ uses oracles $O_p$, then $A$ can make queries to oracles.*
4. *The value of $\mathrm{Exp}^{\mathrm{sr}}_{\Omega}(A, b)$ is $b_A$.*

*The advantage of $A$ is defined as*

$$\mathrm{Adv}^{\mathrm{sr}}_{\Omega}(A)$$

$$= \left|\Pr\left[\mathrm{Exp}^{\mathrm{sr}}_{\Omega}(A,1)=1\right] - \Pr\left[\mathrm{Exp}^{\mathrm{sr}}_{\Omega}(A,0)=1\right]\right|,$$

*and the advantage of* $\Omega$ *is defined as*

$$\mathrm{Adv}^{\mathrm{sr}}_{\Omega}(q) = \max_{A} \mathrm{Adv}^{\mathrm{sr}}_{\Omega}(A)$$

*where* $q$ *is the number of queries to oracle* $O_p$*. If* $\mathrm{Adv}^{\mathrm{sr}}_{\Omega}(q)$ *is not larger than some criterion, then* $\Omega$ *is an* $[n,k,d,\alpha,\beta,B,\ell,\mu]$ *secure regenerating code.*

The find stage is a stage for the adversary to investigate $\Omega$ in advance. Examples of auxiliary information $s$ include the message $m$ and information about tricks embedded in $m$. The guess stage is a stage for the adversary to guess the value of $b$ from incomplete shares, piece vectors, and the auxiliary information. When no oracle $O_p$ exists and the adversary has unbounded computational power, Definition 1 is interpreted as the case of $\mathrm{Adv}^{\mathrm{sr}}_{\Omega}(0) = 0$ (i.e., perfect indistinguishability).

### 5.2 New Construction of Secure Regenerating Codes

Let us consider a composite scheme described in Sect. 3 again (Fig. 2). Unlike the composite scheme $\Omega_{\mathrm{NG}}$ in Sect. 3, a new composite scheme $\Omega_{\mathrm{new}}$ uses the all-or-nothing transform $\Pi_{\mathrm{new}}$ proposed in the Sect. 4.3. The algorithm of $\Omega_{\mathrm{new}}$ is as follows: Given a message $(m_1,\ldots,m_B) \in \{0,1\}^{lB}$ where $m_i \in \{0,1\}^l$, compute a pseudo-message $(z_1,\ldots,z_{B+1}) \in \{0,1\}^{l(B+1)}$ using $\mathcal{E}$ of $\Pi_{\mathrm{new}}$. Consider the pseudo-message as a vector over $\mathbb{F}_{2^l}$, that is, $(z_1,\ldots,z_{B+1}) \in \mathbb{F}_{2^l}^{B+1}$. For the pseudo-message, produce a share $c^{(i)} = (c_1^{(i)},\ldots,c_\alpha^{(i)}) \in \mathbb{F}_{2^l}^{\alpha}$ of node $i$ using any $[n,k,d,\alpha,\beta,B+1]$ linear regenerating code $\mathcal{G}$ over $\mathbb{F}_{2^l}$. The reconstruction and the regeneration are omitted.

The advantage of $\Omega_{\mathrm{new}}$ is that the overhead caused by the security is only one symbol regardless of $B$. Hence, the overhead is negligible as $B$ is sufficiently large. The overhead of previous secure regenerating codes is plural random symbols as mentioned in Sect. 2.4. Even if $B$ is sufficiently large, the overhead cannot be negligible.

The efficiency of $\Omega_{\mathrm{new}}$ depends on the all-or-nothing transform $\Pi_{\mathrm{new}}$ and the linear regenerating code. The time for $\Pi_{\mathrm{new}}$ is roughly two times as long as the time for standard blockcipher modes (i.e., CBC mode). However, since pseudo-message blocks except for the last one can be computed independently, the time for $\Pi_{\mathrm{new}}$ can be reduced with a parallel processing.

This composite scheme $\Omega_{\mathrm{new}}$ is trivially an $[n,k,d,\alpha,\beta,B]$ regenerating code. We will show that $\Omega_{\mathrm{new}}$ is a secure regenerating code in the sense of Definition 4.

#### 5.2.1 MBR Codes

Suppose that $\mathcal{G}$ is the $[n,k,d,\alpha,\beta,B+1]$ linear MBR code. According to experiment $\mathrm{Exp}^{\mathrm{sr}}_{\Omega_{\mathrm{new}}}(A,b)$, we evaluate the advantage of $\Omega_{\mathrm{new}}$.

1. An adversary $A$ chooses $(m_1,\ldots,m_B)$ and $(\hat{m}_1,\ldots,\hat{m}_B)$. Auxiliary information $s$ is generated if necessary.
2. This step is not performed by $A$.
3. Since $d\beta = \alpha$ from Eq. (4), the adversary does not obtain any extra information from a piece vector. Hence, we can assume that $\tau = k-1$ and $\mu = 0$. $A$ makes queries as $c^{(i_1)},\ldots,c^{(i_{k-1})}$ to $O$. $A$ outputs $b_A$ that is determined with $A$'s attacking algorithm.
4. The value of $\mathrm{Exp}^{\mathrm{sr}}_{\Omega_{\mathrm{new}}}(A,b)$ is $b_A$.

Since $A$ knows only $k-1$ shares, $A$ cannot obtain the entire pseudo-message. That is, $A$ does not know at least one pseudo-message symbol or only knows linearly-combined values on unknown pseudo-message symbols. If there exists an adversary $B$ such that

$$\mathrm{Adv}^{\mathrm{sr}}_{\Omega_{\mathrm{new}}}(B) > \mathrm{Adv}^{\mathrm{aon2}}_{\Pi_{\mathrm{new}}}(q), \tag{26}$$

then it contradicts Eq. (17) because the output of $\Omega_{\mathrm{new}}$ is obtained by transforming the output of $\Pi_{\mathrm{new}}$ linearly. Hence,

$$\mathrm{Adv}^{\mathrm{sr}}_{\Omega_{\mathrm{new}}}(q) \le \mathrm{Adv}^{\mathrm{aon2}}_{\Pi_{\mathrm{new}}}(q) \le \frac{2(2B+3)q}{2^l - q}. \tag{27}$$

The discussion above is summarized below.

**Theorem 1.** *When the underlying regenerating code is an* $[n,k,d,\alpha,\beta,B+1]$ *linear MBR code, the composite scheme* $\Omega_{\mathrm{new}}$ *is an* $[n,k,d,\alpha,\beta,B,k-1,\mu]$ *secure regenerating code where* $0 \le \mu \le k-1$*. The advantage of* $\Omega_{\mathrm{new}}$ *is given by*

$$\mathrm{Adv}^{\mathrm{sr}}_{\Omega_{\mathrm{new}}}(q) \le \frac{2(2B+3)q}{2^l - q}. \tag{28}$$

Numerical example: Let us consider conditions for satisfying

$$\frac{2(2B+3)q}{2^l - q} \le 2^{-80} \tag{29}$$

to achieve $\mathrm{Adv}^{\mathrm{sr}}_{\Omega_{\mathrm{new}}}(q) \le 2^{-80}$. Let $l = 128$, which is a typical size supported by well-known blockciphers. Substituting it into Eq. (29) gives

$$q(4B + 6 + 2^{-80}) \le 2^{48}. \tag{30}$$

Since $B \ge 1$, $6 + 2^{-80}$ is not larger than $12B$. The left-hand side of Eq. (30) can be bounded as follows:

$$q(4B + 6 + 2^{-80}) \le q(4B + 12B) \tag{31}$$

Hence, $qB \le 2^{44}$ is a sufficient condition to achieve $\mathrm{Adv}^{\mathrm{sr}}_{\Omega_{\mathrm{new}}}(q) \le 2^{-80}$ for $l = 128$. For instance, suppose that the bit length of a message is 1 [MiB][†] (i.e., $lB = 2^{23}$). When $B = 2^{16}$, $q$ is not larger than $2^{28}$. Thus, when the security level is fixed, there are a trade-off between the message size and the number of queries to encryption/decryption oracles.

Equation (28) is independent of regenerating-code parameters. This fact is caused by the construction that the encoding of a regenerating code is performed after an all-or-nothing transform. Since the indistinguishability has been

[†]MiB means the unit symbol for the *mebibyte*. The binary prefix *mebi* means $2^{20}$.

achieved by the all-or-nothing transform, the subsequent regenerating code has no effect on the indistinguishability.

Suppose that $\mathcal{G}$ is the $[n, k, d, d, 1, k(2d-k+1)/2]$ PM-MBR code that was used in Sect. 2.4. Note that PM-MBR codes are a subset of linear MBR codes. From Theorem 1, the composite scheme $\Omega_{\text{new}}$ is an $[n, k, d, d, 1, k(2d-k+1)/2-1, k-1, \mu]$ secure regenerating code. As mentioned in Sect. 2.4, when $\mathcal{G}$ is used as the underlying regenerating code, the $[n, k, d, d, 1, d-k+1, k-1, \mu]$ PM-SR code can be constructed. Although it is difficult to give a fair comparison of them because of difference of security definitions, the composite scheme $\Omega_{\text{new}}$ is more efficient than the PM-SR code in terms of the number of message symbols because the following inequality holds for $2 \le k \le d$.

$$\frac{k(2d-k+1)}{2} - 1 \ge d - k + 1 \tag{32}$$

For example, let $\mathcal{G}$ be the $[n, 4, 6, 6, 1, 18]$ PM-MBR code that was used in Sect. 2.4. Using $\mathcal{G}$, we obtain

- the $[n, 4, 6, 6, 1, 17, 3, \mu]$ secure regenerating code $\Omega_{\text{new}}$, or
- the $[n, 4, 6, 6, 1, 3, 3, \mu]$ PM-SR code.

Thus, $\Omega_{\text{new}}$ is 17/3 times better than the PM-SR code with respect to the size of a message. Note that the security of $\Omega_{\text{new}}$ is based on the ideal-cipher model whereas the security the PM-SR code is information theoretic.

### 5.2.2 Linear Regenerating Codes

This section discusses the case of linear regenerating codes that includes MBR codes discussed in the previous section. Let $\mathcal{G}$ be the $[n, k, d, \alpha, \beta, B+1]$ linear regenerating code. Suppose that an adversary knows $\tau$ shares $c^{(i_j)}$ of nodes $i_1, \ldots, i_\tau$ and $\mu$ piece vectors for nodes $f_1, \ldots, f_\mu$. Owing to the linear reconstruction and the linear regeneration (see Sect. 2.3), the adversary obtains the following system of equations with respect to a pseudo-message $(z_1, \ldots, z_{B+1})$.

$$\begin{pmatrix} c^{(i_1)} \\ \vdots \\ c^{(i_\tau)} \\ d_{f_1}^{(i_{f_1,1},\ldots,i_{f_1,d})} \\ \vdots \\ d_{f_\mu}^{(i_{f_\mu,1},\ldots,i_{f_\mu,d})} \end{pmatrix} = \mathbf{M}_{\text{a}} \begin{pmatrix} z_1 \\ \vdots \\ z_{B+1} \end{pmatrix} \text{ over } \mathbb{F}_{2^l} \tag{33}$$

where $\mathbf{M}_{\text{a}}$ is a matrix over $\mathbb{F}_{2^l}$ that depends on compromised shares and piece vectors. Since the length of the vector $c^{(i_1)}$ is $\alpha$ and that of the vector $d_{f_1}^{(i_{f_1,1},\ldots,i_{f_1,d})}$ is $d\beta$, the size of $\mathbf{M}_{\text{a}}$ is $(\tau\alpha + \mu d\beta) \times (B+1)$. The matrix $\mathbf{M}_{\text{a}}$ is known to the adversary because the adversary knows the reconstructing algorithm and the regenerating algorithm of $\mathcal{G}$.

Suppose that $\tau\alpha + \mu d\beta \le B$. Since our objective is to evaluate the advantage of an adversary, we have only to consider that $\tau\alpha + \mu d\beta = B$ and $B$ rows of $\mathbf{M}_{\text{a}}$ are linear independent. Hence, Eq. (33) is the same structure as Eq. (15) by considering $\mathbf{M}_{\text{a}}$ as $\mathbf{U}$.

**Theorem 2.** *Suppose that $\mathcal{G}$ is an $[n, k, d, \alpha, \beta, B+1]$ linear regenerating code. The composite scheme $\Omega_{\text{new}}$ is an $[n, k, d, \alpha, \beta, B, \ell, \mu]$ secure regenerating code where $0 \le \mu \le \ell$ if*

$$(\ell - \mu)\alpha + \mu d\beta \le B. \tag{34}$$

*The advantage of $\Omega_{\text{new}}$ is given by*

$$\text{Adv}^{\text{sr}}_{\Omega_{\text{new}}}(q) \le \frac{2(2B+3)q}{2^l - q}.$$

First, suppose that $\mathcal{G}$ is an $[n, k, d, \alpha, \beta, B+1]$ linear MBR code. Since $\alpha = d\beta$ because of Eq. (4), Eq. (34) results in

$$(\ell - \mu)d\beta + \mu d\beta \le B.$$

Substituting Eq. (4) into it yields

$$\frac{2\ell d(B+1)}{k(2d-k+1)} \le B$$

because $\mathcal{G}$ is an $[n, k, d, \alpha, \beta, B+1]$ linear regenerating code. This inequality can be transformed into

$$\ell \le k \cdot \frac{B}{B+1}\left(1 - \frac{k-1}{2d}\right). \tag{35}$$

When the inequality above holds, the composite scheme $\Omega_{\text{new}}$ is the secure regenerating code. On the other hand, the condition in Theorem 1 is

$$\ell \le k - 1. \tag{36}$$

Comparing Eq. (35) with Eq. (36), we observe that Theorem 1 is of wider application than Theorem 2 because Eq. (36) does not depend on $B, d$.

Numerical example: Let us consider the same numerical example as Theorem 1. In addition to $l = 128$, $B = 2^{16}$ and $q \le 2^{28}$, suppose that $k = 10$ and $d = 11$. Compare Theorem 2 with Theorem 1 in terms of the acceptable number of compromised nodes (i.e., $\ell$) to achieve the security of $2^{-80}$. In the case of Theorem 2, $\ell$ has to satisfy

$$\ell \le 10 \cdot \frac{2^{16}}{2^{16}+1}\left(1 - \frac{9}{22}\right) \approx 5.90.$$

On the other hand, in the case of Theorem 1, $\ell$ has to satisfy that $\ell \le 9$.

Next, suppose that $\mathcal{G}$ is an $[n, k, d, \alpha, \beta, B+1]$ linear MSR code. From Eq. (5), Eq. (34) is written as

$$\frac{(\ell - \mu)(B+1)}{k} + \frac{\mu d(B+1)}{k(d-k+1)} \le B,$$

namely,

$$\frac{\ell - \mu}{k} + \frac{\mu d}{k(d-k+1)} \le \frac{B}{B+1}. \tag{37}$$

Suppose that $\mathcal{G}$ is the $[n, k, 2(k-1), k-1, 1, k(k-1)]$ PM-MSR code that was used in Sect. 2.4. Substituting parameters into Eq. (37) gives

$$\ell + \mu \le k - \frac{1}{k-1}. \tag{38}$$

When the inequality above satisfies, the composite scheme $\Omega_{\text{new}}$ is an $[n, k, 2(k-1), k-1, 1, k(k-1)-1, \ell, \mu]$ secure regenerating code. As mentioned in Sect. 2.4, given $\mathcal{G}$, the $[n, k, 2(k-1), k-1, 1, (k-1)(k-2), \ell, \mu]$ PM-SR code can be constructed if

$$\ell + \mu \le 2. \tag{39}$$

Although it is difficult to give a fair comparison of them because of difference of security definitions, $\Omega_{\text{new}}$ seems to be better than the PM-SR code for two reasons.

1. The number of message symbols is larger. The number of message symbols of $\Omega_{\text{new}}$ is $k(k-1)-1$, and that of the PM-SR code is $(k-1)(k-2)$. For $k \ge 2$, the following inequality holds.

   $$k(k-1) - 1 \ge (k-1)(k-2).$$

2. The condition on $\ell, \mu$ is relaxed. Comparing Eq. (38) with Eq. (39), we see that

   $$k - \frac{1}{k-1} \ge 2$$

   for $k \ge 3$.

For example, let $\mathcal{G}$ be the $[n, 5, 8, 4, 1, 20]$ PM-MSR code that was used in Sect. 2.4. Using $\mathcal{G}$, we obtain

- the $[n, 5, 8, 4, 1, 19, \ell, \mu]$ secure regenerating code $\Omega_{\text{new}}$ for $\ell + \mu \le 19/4$, or
- the $[n, 5, 8, 4, 1, 12, \ell, \mu]$ PM-SR code for $\ell + \mu \le 2$.

The upper bound of the secrecy capacity of secure MSR codes and the number of secret message symbols of a PM-SR code using a PM-MSR code have been recently shown in articles [13], [14], respectively. These codes achieve information-theoretical security. Since the number of message symbols of the proposed composite scheme equals that of an underlying MSR code minus one, the proposed composite scheme is more efficient than these codes in terms of the number of message symbols.

## 6. Proof of Lemma 1

### 6.1 Overview

When two messages to be distinguished are given by an adversary, we notice the following two facts

- If the adversary does not make any queries with all keys $\kappa \parallel i$, then pseudo-message symbols $z_1, \ldots, z_B$ are random strings because of the ideal-cipher model.
- If the adversary does not make queries with a some key $z_{i'} \parallel (B + i')$, then the symbol $w_{i'}$ is a random string because of the ideal-cipher model. It follows that $z_{B+1}$ is a random string from the viewpoint of the adversary.

If the adversary does not make such queries, then the advantage of the adversary is precisely 0, that is,

$$\Pr\left[\mathrm{Exp}^{\mathrm{aon2}}_{\Pi}(A, b) = b\right] = \frac{1}{2}.$$

Hence, we primarily evaluate probabilities of the following events:

- the adversary makes some query with $\kappa \parallel *$ where '$*$' denotes any integer,
- the adversary makes some query with $z_{i'} \parallel (B + i')$ where $z_{i'}$ is unknown to the adversary.

### 6.2 Proof

Let $D$ be the decryption of the ideal cipher and let $\mathsf{Qry}(\chi)$ denote the event that $A$ makes a query such that a key is $\chi$ (i.e., $E(\chi, \cdot)$ or $D(\chi, \cdot)$). Let $q_{\mathsf{f}}$ and $q_{\mathsf{g}}$ be the number of queries to the ideal cipher in the find stage and that in the guess stage, respectively. We assume that $A$ does not make a query such that the answer is determined by previous queries.

Let $\mathsf{Bad}_{\mathsf{f}}$ be the event in the find stage such that

$$\begin{aligned}\mathsf{Bad}_{\mathsf{f}} = &(\mathsf{Qry}(\kappa \parallel 1) \vee \cdots \vee \mathsf{Qry}(\kappa \parallel B)) \\ &\vee (\mathsf{Qry}(z_1 \parallel (B+1)) \vee \cdots \vee \mathsf{Qry}(z_B \parallel (B+B))).\end{aligned}$$

This case is that one pseudo-message symbol $z_\iota$ is unknown and $B$ pseudo-message symbols are known to $A$. Let $\mathsf{Bad}_{\mathsf{g}}$ be the event in the guess stage such that

$$\begin{aligned}\mathsf{Bad}_{\mathsf{g}} = &(\mathsf{Qry}(\kappa \parallel 1) \vee \cdots \vee \mathsf{Qry}(\kappa \parallel B)) \\ &\vee \mathsf{Qry}(z_\iota \parallel (B + \iota))\end{aligned}$$

where $z_\iota$ is a missing symbol in the guess stage. Let $\mathsf{Bad}_{\mathsf{fg}}$ be an event as

$$\mathsf{Bad}_{\mathsf{fg}} = \mathsf{Bad}_{\mathsf{f}} \vee \mathsf{Bad}_{\mathsf{g}}$$

and let $\mathsf{Suc}$ denote the event of $\mathrm{Exp}^{\mathrm{aon2}}_{\Pi_{\text{new}}}(A, b) = b$ for abbreviation.

Denoting by $\overline{\mathsf{E}}$ the complementary event of an event $\mathsf{E}$, we have

$$\begin{aligned}
&\Pr\left[\mathrm{Exp}^{\mathrm{aon2}}_{\Pi_{\text{new}}}(A, b) = b\right] \\
&= \Pr[\mathsf{Suc}] && (40)\\
&= \Pr\left[\mathsf{Suc}|\overline{\mathsf{Bad}_{\mathsf{fg}}}\right]\Pr\left[\overline{\mathsf{Bad}_{\mathsf{fg}}}\right] + \Pr\left[\mathsf{Suc}|\mathsf{Bad}_{\mathsf{fg}}\right]\Pr\left[\mathsf{Bad}_{\mathsf{fg}}\right] \\
&\le \Pr\left[\mathsf{Suc}|\overline{\mathsf{Bad}_{\mathsf{fg}}}\right] + \Pr\left[\mathsf{Bad}_{\mathsf{fg}}\right] && (41)\\
&\le \Pr\left[\mathsf{Suc}|\overline{\mathsf{Bad}_{\mathsf{fg}}}\right] + \Pr[\mathsf{Bad}_{\mathsf{f}}] + \Pr\left[\mathsf{Bad}_{\mathsf{g}}\right] && (42)\\
&\le \Pr\left[\mathsf{Suc}|\overline{\mathsf{Bad}_{\mathsf{fg}}}\right] + \Pr[\mathsf{Bad}_{\mathsf{f}}] \\
&\quad + \Pr\left[\mathsf{Bad}_{\mathsf{g}}|\mathsf{Bad}_{\mathsf{f}}\right]\Pr[\mathsf{Bad}_{\mathsf{f}}] + \Pr\left[\mathsf{Bad}_{\mathsf{g}}|\overline{\mathsf{Bad}_{\mathsf{f}}}\right]\Pr\left[\overline{\mathsf{Bad}_{\mathsf{f}}}\right] \\
&\le \Pr\left[\mathsf{Suc}|\overline{\mathsf{Bad}_{\mathsf{fg}}}\right] + 2\Pr[\mathsf{Bad}_{\mathsf{f}}] + \Pr\left[\mathsf{Bad}_{\mathsf{g}}|\overline{\mathsf{Bad}_{\mathsf{f}}}\right]. && (43)
\end{aligned}$$

Inequalities (41)–(43) are because

$$\text{Eq. (41): } \Pr\left[\overline{\mathsf{Bad}_{\mathsf{fg}}}\right] \le 1,\ \Pr\left[\mathsf{Suc}|\mathsf{Bad}_{\mathsf{fg}}\right] \le 1,$$
$$\text{Eq. (42): } \Pr\left[\mathsf{Bad}_{\mathsf{fg}}\right] = \Pr\left[\mathsf{Bad}_{\mathsf{f}} \vee \mathsf{Bad}_{\mathsf{g}}\right] \le \Pr\left[\mathsf{Bad}_{\mathsf{f}}\right] + \Pr\left[\mathsf{Bad}_{\mathsf{g}}\right],$$
$$\text{Eq. (43): } \Pr\left[\mathsf{Bad}_{\mathsf{g}}|\mathsf{Bad}_{\mathsf{f}}\right] \le 1,\ \Pr\left[\overline{\mathsf{Bad}_{\mathsf{f}}}\right] \le 1.$$

We evaluate three terms in Eq. (43) in sequence.

(1) First term: $\Pr\left[\mathsf{Suc}|\overline{\mathsf{Bad}_{\mathsf{fg}}}\right]$

Suppose that $b = 0$. In the ideal cipher model, the encryption (or the decryption) is an independent random permutation for a different key and the answer of the first query is uniformly chosen from $\{0,1\}^l$ at random. Hence, each $z_i$ ($i = 1, \ldots, B$) is uniformly and independently chosen from $\{0,1\}^l$ at random. That is, it is identical to $(z_1, \ldots, z_B) \twoheadleftarrow \{0,1\}^{lB}$, which is the operation when $b = 1$. We have

$$\Pr\left[\mathsf{Suc}|\overline{\mathsf{Bad}_{\mathsf{fg}}}\right] = \frac{1}{2}. \tag{44}$$

(2) Second term: $\Pr\left[\mathsf{Bad}_{\mathsf{f}}\right]$

All the queries in the find stage are made before $\kappa$ and $z_i$ are determined. It follows that the best strategy for maximizing $\Pr\left[\mathsf{Bad}_{\mathsf{f}}\right]$ is to make $q_{\mathsf{f}}$ queries with distinct keys. Let $\mathsf{Bad}_{\mathsf{f}}^j$ be the event that the key of the $j$-th query is $\kappa \parallel i$ or $z_i \parallel (B+i)$ where $1 \le \forall i \le B$. We have

$$\begin{aligned}
&\Pr\left[\mathsf{Bad}_{\mathsf{f}}\right] \\
&\le \Pr\left[\mathsf{Bad}_{\mathsf{f}}^1\right] + \sum_{j=2}^{q_{\mathsf{f}}} \Pr\left[\mathsf{Bad}_{\mathsf{f}}^j|\overline{\mathsf{Bad}_{\mathsf{f}}^{j-1}}\right] \\
&\le \frac{B+1}{2^l} + \sum_{j=2}^{q_{\mathsf{f}}} \frac{B+1}{2^l-(j-1)} \\
&\le \frac{(B+1)q_{\mathsf{f}}}{2^l - q_{\mathsf{f}}}.
\end{aligned} \tag{45}$$

Note that $\Pr\left[\mathsf{Bad}_{\mathsf{f}}^1\right] \le (B+1)/2^l$ because a part of the key is the known number $i \in \{1, \ldots, 2B\}$.

(3) Last term: $\Pr\left[\mathsf{Bad}_{\mathsf{g}}|\overline{\mathsf{Bad}_{\mathsf{f}}}\right]$

Let $\mathsf{Last}$ be the event that the missing symbol $z_\iota$ is $z_{B+1}$. The last term is bounded by

$$\begin{aligned}
&\Pr\left[\mathsf{Bad}_{\mathsf{g}}|\overline{\mathsf{Bad}_{\mathsf{f}}}\right] \\
&= \Pr\left[\mathsf{Bad}_{\mathsf{g}}|\overline{\mathsf{Bad}_{\mathsf{f}}} \wedge \mathsf{Last}\right]\Pr\left[\mathsf{Last}\right] \\
&\quad + \Pr\left[\mathsf{Bad}_{\mathsf{g}}|\overline{\mathsf{Bad}_{\mathsf{f}}} \wedge \overline{\mathsf{Last}}\right]\Pr\left[\overline{\mathsf{Last}}\right] \\
&\le \Pr\left[\mathsf{Bad}_{\mathsf{g}}|\overline{\mathsf{Bad}_{\mathsf{f}}} \wedge \mathsf{Last}\right] + \Pr\left[\mathsf{Bad}_{\mathsf{g}}|\overline{\mathsf{Bad}_{\mathsf{f}}} \wedge \overline{\mathsf{Last}}\right]
\end{aligned} \tag{46}$$

First, suppose that $\mathsf{Last}$ occurs. Then, the adversary cannot obtain any information about the pseudo-key $\kappa$. Let $\mathsf{Bad}_{\mathsf{g}}^j$ be the event that the key of the $j$-th query is $\kappa \parallel i$ where $1 \le \forall i \le B$. Then,

$$\begin{aligned}
&\Pr\left[\mathsf{Bad}_{\mathsf{g}}|\overline{\mathsf{Bad}_{\mathsf{f}}} \wedge \mathsf{Last}\right] \\
&\le \Pr\left[\mathsf{Bad}_{\mathsf{g}}^1|\overline{\mathsf{Bad}_{\mathsf{f}}} \wedge \mathsf{Last}\right] + \sum_{j=2}^{q_{\mathsf{g}}} \Pr\left[\mathsf{Bad}_{\mathsf{g}}^j|\overline{\mathsf{Bad}_{\mathsf{g}}^{j-1}} \wedge \mathsf{Last}\right] \\
&\le \frac{1}{2^l - q_{\mathsf{f}}} + \sum_{j=2}^{q_{\mathsf{g}}} \frac{1}{2^l - (q_{\mathsf{f}} + j - 1)} \\
&\le \frac{q_{\mathsf{g}}}{2^l - (q_{\mathsf{f}} + q_{\mathsf{g}})}.
\end{aligned} \tag{47}$$

Next, suppose that $\mathsf{Last}$ does not occur. Let $\mathsf{Bad}_{\mathsf{g}}^j$ be the event that the key of the $j$-th query is $\kappa \parallel i$ where $1 \le \forall i \le B$ or $z_\iota \parallel (B+\iota)$. Then,

$$\begin{aligned}
&\Pr\left[\mathsf{Bad}_{\mathsf{g}}|\overline{\mathsf{Bad}_{\mathsf{f}}} \wedge \overline{\mathsf{Last}}\right] \\
&\le \Pr\left[\mathsf{Bad}_{\mathsf{g}}^1|\overline{\mathsf{Bad}_{\mathsf{f}}} \wedge \overline{\mathsf{Last}}\right] + \sum_{j=2}^{q_{\mathsf{g}}} \Pr\left[\mathsf{Bad}_{\mathsf{g}}^j|\overline{\mathsf{Bad}_{\mathsf{g}}^{j-1}} \wedge \overline{\mathsf{Last}}\right] \\
&\le \frac{2}{2^l - q_{\mathsf{f}}} + \sum_{j=2}^{q_{\mathsf{g}}} \frac{2}{2^l - (q_{\mathsf{f}} + j - 1)} \\
&\le \frac{2q_{\mathsf{g}}}{2^l - (q_{\mathsf{f}} + q_{\mathsf{g}})}.
\end{aligned} \tag{48}$$

Substituting Eq. (47) and Eq. (48) into Eq. (46) gives

$$\begin{aligned}
\Pr\left[\mathsf{Bad}_{\mathsf{g}}|\overline{\mathsf{Bad}_{\mathsf{f}}}\right] &\le \frac{q_{\mathsf{g}}}{2^l - (q_{\mathsf{f}} + q_{\mathsf{g}})} + \frac{2q_{\mathsf{g}}}{2^l - (q_{\mathsf{f}} + q_{\mathsf{g}})} \\
&= \frac{3q_{\mathsf{g}}}{2^l - (q_{\mathsf{f}} + q_{\mathsf{g}})}.
\end{aligned} \tag{49}$$

Substituting Eq. (44), Eq. (45), and Eq. (49) into Eq. (43) yields

$$\begin{aligned}
\Pr\left[\mathsf{Suc}\right] &\le \frac{1}{2} + \frac{2(B+1)q_{\mathsf{f}}}{2^l - q_{\mathsf{f}}} + \frac{3q_{\mathsf{g}}}{2^l - (q_{\mathsf{f}} + q_{\mathsf{g}})} \\
&\le \frac{1}{2} + \frac{2(B+1)q_{\mathsf{f}} + 3q_{\mathsf{g}}}{2^l - (q_{\mathsf{f}} + q_{\mathsf{g}})} \quad (\because q_{\mathsf{g}} \ge 0) \\
&\le \frac{1}{2} + \frac{2Bq_{\mathsf{f}} + 3(q_{\mathsf{f}} + q_{\mathsf{g}})}{2^l - (q_{\mathsf{f}} + q_{\mathsf{g}})} \quad (\because q_{\mathsf{f}} \ge 0) \\
&\le \frac{1}{2} + \frac{(2B+3)(q_{\mathsf{f}} + q_{\mathsf{g}})}{2^l - (q_{\mathsf{f}} + q_{\mathsf{g}})} \quad (\because q_{\mathsf{g}} \ge 0).
\end{aligned}$$

Letting $q = q_{\mathsf{f}} + q_{\mathsf{g}}$, we have

$$\Pr\left[\mathsf{Suc}\right] \le \frac{1}{2} + \frac{(2B+3)q}{2^l - q}. \tag{50}$$

Substituting the inequality above into Eq. (16) gives

$$\begin{aligned}
\mathrm{Adv}^{\mathrm{aon2}}_{\Pi_{\mathrm{new}}}(A) &= \left|2\Pr\left[\mathrm{Exp}^{\mathrm{aon2}}_{\Pi_{\mathrm{new}}}(A,b) = b\right] - 1\right| \\
&= \left|2\Pr\left[\mathsf{Suc}\right] - 1\right| \quad (\because \text{Eq. (40)}) \\
&\le \frac{2(2B+3)q}{2^l - q}. \quad (\because \text{Eq. (50)})
\end{aligned}$$

Since the analysis above does not depend on the algorithm of the adversary $A$, we obtain the following inequation from Eq. (16).

$$\mathrm{Adv}^{\mathrm{aon2}}_{\Pi_{\mathrm{new}}}(q) = \max_{A} \mathrm{Adv}^{\mathrm{aon2}}_{\Pi_{\mathrm{new}}}(A) \leq \frac{2(2B+3)q}{2^l - q}.$$

## 7. Concluding Remarks

Although previous secure regenerating codes are unconditionally secure against eavesdropping, the confidentiality caused large overhead. Since it is impossible to decrease the overhead as keeping information-theoretic security, we introduced the new definition of secure regenerating codes using the concept of the indistinguishability.

We have proposed secure regenerating codes satisfying the new definition. The proposed secure regenerating code consists of a new all-or-nothing transform and the non-secure linear regenerating code. The overhead is negligibly small as the message size is sufficiently large. In fact, the overhead of the proposed codes is less than that of previous secure regenerating codes.

Showing the counter example, we explained that the previous definition of the all-or-nothing transform is insufficient for achieving secure regenerating codes that satisfy the new definition. We have relaxed the definition of an all-or-nothing transform and have shown the new all-or-nothing transform satisfying the relaxed definition.

The security analysis of the proposed codes is based on the ideal cipher model. When the proposed code is implemented, the ideal cipher has to be instantiated by a practical cipher such as AES because the ideal cipher is a mathematical abstraction used in cryptographic proofs. Hence, our security analysis can be regarded as the security analysis when an adversary considers the practical cipher as a black box.

As is well known, regenerating codes on which this paper focuses are a concept born of network coding. Network coding which is information-theoretically secure against eavesdropping (called secure network coding) has been studied (e.g., [15]–[18]). Using the idea of this paper, we can probably construct efficient secure network coding from any network coding, but we leave this as a future work. We also leave the implementation of our secure regenerating code.

## Acknowledgments

The authors would like to thank reviewers for useful comments. This is a product of research which was financially supported in part by the Kansai University Outlay Support for Establish Research Centers, 2015 “Further development of identification technology based on device fingerprints.”

### References

[1] A.G. Dimakis, P.B. Godfrey, Y. Wu, M.J. Wainwright, and K. Ramchandran, “Network coding for distributed storage systems,” IEEE Trans. Inf. Theory, vol.56, no.9, pp.4539–4551, Sept. 2010.

[2] K.V. Rashmi, N.B. Shah, and P.V. Kumar, “Optimal exact-regenerating codes for distributed storage at the MSR and MBR points via a product-matrix construction,” IEEE Trans. Inf. Theory, vol.57, no.8, pp.5227–5239, 2011.

[3] S. Pawar, S.E. Rouayheb, and K. Ramchandran, “On secure distributed data storage under repair dynamics.” CoRR, http://arxiv.org/abs/1003.0488, 2010.

[4] N.B. Shah, K.V. Rashmi, and P.V. Kumar, “Information-theoretically secure regenerating codes for distributed storage,” Global Telecommunications Conference (GLOBECOM 2011), pp.1–5, 2011.

[5] M. Kurihara and H. Kuwakado, “Secure regenerating codes based on Rashmi-Shah-Kumar MBR codes,” IEICE Trans. Fundamentals, vol.E96-A, no.2, pp.635–648, Feb. 2013.

[6] M. Kurihara and H. Kuwakado, “Generalization of Rashmi-Shah-Kumar minimum-storage-regenerating codes,” CoRR, abs/1309.6701, 2013. http://arxiv.org/abs/1309.6701.

[7] M. Kurihara and H. Kuwakado, “Secure regenerating codes based on MSR codes for distributed storage systems,” IEICE Trans. Fundamentals (Japanese Edition), vol.J96-A, no.4, pp.166–174, April 2013.

[8] J. Kurihara and Y. Miyake, “Securing distributed storage systems based on arbitrary regenerating codes,” IEICE Commun. Express, vol.2, no.10, pp.442–446, 2013.

[9] R.L. Rivest, “All-or-nothing encryption and the package transform,” Fast Software Encryption FSE ’97, Lecture Notes in Computer Science, vol.1267, pp.210–218, 1997.

[10] S.-J. Lin and W.-H. Chung, “An unified form of exact-MSR codes via product-matrix framework,” 2013 IEEE 24th International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC), pp.830–834, Sept. 2013.

[11] N.B. Shah, K.V. Rashmi, P.V. Kumar, and K. Ramchandran, “Distributed storage codes with repair-by-transfer and nonachievability of interior points on the storage-bandwidth tradeoff,” IEEE Trans. Inf. Theory, vol.58, no.3, pp.1837–1852, March 2012.

[12] A. Desai, “The security of all-or-nothing encryption: Protecting against exhaustive key search,” Advances in Cryptology - CRYPTO 2000, Lecture Notes in Computer Science, vol.1880, pp.359–375, 2000.

[13] B. Sasidharan, P.V. Kumar, N. Shah, K. Rashmi, and K. Ramachandran, “Optimality of the product-matrix construction for secure MSR regenerating codes,” 2014 6th International Symposium on Communications, Control and Signal Processing (ISCCSP), pp.10–14, May 2014.

[14] K. Huang, U. Parampalli, and M. Xian, “Characterization of secrecy capacity for general MSR codes under passive eavesdropping model,” CoRR, vol.abs/1505.01986, 2015. http://arxiv.org/abs/1505.01986.

[15] N. Cai and R.W. Yeung, “Secure network coding,” 2002 IEEE International Symposium on Information Theory, p.323, 2002.

[16] J. Feldman, T. Malkin, R.A. Servedio, and C. Stein, “On the capacity of secure network coding,” 42nd Annual Allerton Conference on Communication, Control, and Computing, 2004.

[17] K. Bhattad and K.R. Narayanan, “Weakly secure network coding,” First Workshop on Network Coding, Theory, and Applications, NETCOD 2005, April 2005. http://www.tamu.edu/commtheory/.

[18] D. Silva and F.R. Kschischang, “Universal secure network coding via rank-metric codes,” IEEE Trans. Inf. Theory, vol.57, no.2, pp.1124–1135, Feb. 2011.

**Hidenori Kuwakado** received the B.E., M.E. and D.E. degrees from Kobe University in 1990, 1992, and 1999 respectively. He worked for Nippon Telegraph and Telephone Corporation from 1992 to 1996. From 1996 to 2002, he was a research associate in the Faculty of Engineering, Kobe University. From 2002 to 2007, he was an associate professor in the Faculty of Engineering, Kobe University. From 2007 to 2013, he was an associate professor in Graduate School of Engineering, Kobe University. Since 2013, he has been a professor in Faculty of Informatics, Kansai University. His research interests are in cryptography and information security.

**Masazumi Kurihara** received the B.S. degree in Mathematics from Tokyo Metropolitan University in 1990, and the M.E. and Ph.D. degrees in Computer Science and Information Mathematics from the University of Electro-Communications (UEC) in 1992 and 2002, respectively. From 1992 to 2007, he was a research associate in the Faculty of Electro-Communications at UEC. From 2007 to 2010, he was an assistant professor in the Faculty of Electro-Communications at UEC. Since 2010, he was an assistant professor in the Graduate School of Informatics and Engineering at UEC. His research interests are in algebraic coding theory.