# Improved Solution of Topology Concealment Problem in Computing Network Reliability

Raku Hatanaka[1] and Masahiro Hayashi[2]
*Major of Information, Graduate School of Integrative Science and Engineering*
*Tokyo City University*
1-28-1 Tamazutsumi, Setagaya-ku, Tokyo 158-8557, Japan
E-mail: [1]g2281452@tcu.ac.jp, [2]mhaya@tcu.ac.jp

***Abstract*— This paper proposes an improved method to solve the topology concealment problem in the secret computation to be used in computing network reliability. The key idea is to add dummy links with reliabilities of zeros and ones, while the existing method adds only dummy links with reliabilities of ones. Our proposal realizes stronger security than the existing method without increasing the number of dummy links. A numerical test shows the effectiveness of our proposal.**

***Keywords*— *reliability computation*, *secrete computation*, *communications networks*, *graph*, *topology*, *security***

## I. Introduction

A very important issue in the QoS engineering field is how to build a highly reliable network. Sufficient reliability can be realized by repeating the following steps.

Step 1. Enumerate feasible design plans for networks. (especially our concern here is communications networks).

Step 2. Compute their network reliabilities.

Step 3. Select the most cost effective one satisfying the requested reliability specification.

Step 4. Implement the selected design plan and feedback the findings from it to Steps 1- 3.

In practice, Step 2 is the toughest of these steps for the designer. Thus, determining how to reduce the burden of step 2 is an important problem.

While a possible solution would be outsourcing Step 2 to a subcontractor, this is problematic because the reliabilities of the components, implying the probabilities of such as routers, cables, multiplexers, and others not failing used in this computation are secret information: they should not be revealed to the subcontractor. Accordingly, a secret computation becomes necessary to ensure that the outsourcing computation does not entail revealing such data.

While there have been many studies on secret computation [1]-[5], almost none of them can be applied to the network reliability computation because the standard secret computation scheme involving operations on numbers with numerous digits, such as 0.9999999879 × 0.999999938 + ..., becomes intolerably long [1]-[3][5] or imposes requirements on the outsourcing side and subcontractor to share complete knowledge of the computation process [4].

The *IH* method [6] was proposed to solve this problem, and ref. [7] improved its security strength. In particular, Ref. [6] found a relatively simple technique to perform the secret computation. While it is only applicable to polynomial computations, it causes no problem when we apply it to network reliability, which is obtained using only polynomial computations.

However, ref. [8] pointed out that the methods of refs. [6][7] have a problem when they are used in a network reliability computation, because they require the topology of network to be known to the subcontractor: the topology is a more sensitive form of information than the reliabilities of the components. From this viewpoint, ref. [8] proposed a 'topology concealment problem' on outsourcing the computation of network reliability without revealing information on the topology.

Ref. [8] also gave a solution to the problem. This solution adds dummy links without changing the network reliability so that a hacker has trouble determining the original topology. The outsourcing side can still determine the network reliability because adding dummy links does not change the network reliability. In particular, the dummy link in Ref. [8] has a reliability of one after a single node is split into two.

In this paper, we devise another dummy link addition technique wherein the dummy link has a reliability of zero and the node is not split; this makes it difficult for a hacker to determine the original topology because s/he must identify not only which links are dummies, but also their type. We describe the technique to add link with its reliability being zero without splitting nodes into two. We also discuss the security strength of our proposal and present a numerical test showing the effectiveness of our proposal.

## II. Preparation

Suppose that $f$ is a multivariable polynomial function whose input variables are $x_1, x_2, \dots , x_n$. We will sometimes write $f$ as $f(x_1, x_2, \dots , x_n)$. We define the single variable polynomial $g$ obtained from $f$ by replacing every $x_i$ ($i = 1, 2, \dots , n$) with $x$. We define the 'degree of $f$' to be degree of $g$. For example, if $f = (1- x_1)^2x_2^2 + x_3^2$ then $g = (1- x)^2x^2 + x^2 = 2x^2 - 2x^3 + x^4$ so the degree of $g$ and $f$ is 4. We denote the degree by $m$.

We suppose a scheme equivalent to a common key system. Therefore, its security strength is determined by the difficulty of finding the correct answer (key) from a list of the candidates. For example, if the key is a 3 digit number, then the number of candidates is 1000 ,while if the key is a 4 digit number, the number of candidates is 10000. The latter key is stronger in security than the former.

The security strength is expressed numerically by $\log_2$(the number of candidates): we call this value the 'bits of security'. For example, if the number of candidates is 1000, then the key offers $\log_2(1000) = 9.966$ bits of security.

A graph is a mathematical object consisting nodes and links. An example of a graph is shown in Fig. 1.

Links are identified by natural numbers. One node in the graph is called the source and another is called the sink. These

978-1-6654-9927-9/22/$31.00 ©2022 IEEE

are illustrated by filled circles in the figures of this paper. The word 'topology' is used to refer to a graph when we discuss about adding links to it.

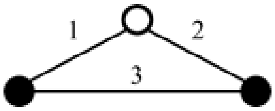

Fig. 1. Example of a graph.

Links are identified by natural numbers. One node in the graph is called the source and another is called the sink. These are illustrated by filled circles in the figures of this paper. The word 'topology' is used to refer to a graph when we discuss about adding links to it.

We add links to change the topology. Each added link is called a dummy link. There are two types of dummy link. One, called a complete dummy link, is added when a single node is split into two. The other, called a zero dummy link, is added without splitting the node. A complete dummy link never fails while a zero dummy link always fails. $v$ denotes the number of links of the topology before adding dummy links, and $L$ is the number of added dummy links. TThe total number of links of the topology is $v + L$.

Note that the definition of bits of security in this paper is a slightly different from the commonly used definition in ref. [9]. However, we will use it here in continuity with refs. [6]-[8].

## III. Fully Homomorphic Encryption

This section summarizes fully homomorphic encryption (FHE) [1]-[7], which is the main way of realizing secret computations.

### A. Basic Idea of FHE

This form of encryption is realized by a mapping *Enc*( ) having the following three properties, where $\alpha$ and $\beta$ are numbers.

Property 1. $Enc(\alpha + \beta) = Enc(\alpha) + Enc(\beta)$
Property 2. $Enc(\alpha \times \beta) = Enc(\alpha) \times Enc(\beta)$
Property 3. $Enc^{-1}$ can be generated only by the persons who know key.

Let us consider the case of outsourcing the computation 2 × (1 + 5). We can obtain the result 12 by outsourcing $Enc(2) \times (Enc(1) + Enc(5))$ instead of outsourcing 2 × (1 + 5) if we know the key, because we find that $Enc(12) = Enc(2 \times (1 + 5)) = Enc(2) \times (Enc(1) + Enc(5))$ from Properties 1 and 2 and we can obtain 12 from $12 = Enc^{-1}(Enc(2) \times (Enc(1) + Enc(5)))$ by using Property 3.

The scheme using a mapping satisfying Properties 1, 2 and 3 is called fully homomorphic encryption, and many researchers have studied it after the pioneering work of Gentry [1][2].

### B. IH Method

The *IH* method was recently proposed as a way of realizing *FHE* [6][7]. It is the only method that can be used for outsourcing the computation of network reliability, as explained in Subsection *B* of Section IV.

Here, suppose that $f$ is a function expressed by a multivariate polynomial with input variables $x_1, x_2, \ldots, x_n$ as defined in Section II. The *IH* method encrypts these input variables using real numbers $H_1, H_2, \ldots, H_{m+1}, \lambda_1, \lambda_2, \ldots, \lambda_n$ to the following.

ENCRYPTION:

$$x_1 + H_1\lambda_1, x_2 + H_1\lambda_2, \ldots, x_n + H_1\lambda_n$$
$$x_1 + H_2\lambda_1, x_2 + H_2\lambda_2, \ldots, x_n + H_2\lambda_n$$
$$\ldots.$$
$$x_1 + H_{m+1}\lambda_1, x_2 + H_{m+1}\lambda_2, \ldots, x_n + H_{m+1}\lambda_n$$

In the above, $H_1 \neq 0, H_2 \neq 0, \ldots, H_{m+1} \neq 0, \lambda_1 \neq 0, \lambda_2 \neq 0, \ldots, \lambda_n \neq 0$, where $m$ is defined in Section II.

Ref. [6] proved the following theorem by using a special mapping defined in ref. [10] satisfying Properties 1 and 2 in the previous subsection.

Theorem. Let $F$ be an $(m+1) \times (m+1)$ matrix in which each element $(a, b)$ is of the form $\frac{H_a{}^{b-1}}{(b-1)!}$. Furthermore, let $G$ be an $m + 1$ column vector whose $a$-th element is $f(x_1\lambda_1 + H_a, x_2 + H_a\lambda_2, \ldots, x_n + H_a\lambda_n)$. The first element of the column vector obtained by $F^{-1}G$ equals $f$, where $F^{-1}$ is the inverse matrix of $F$.

This theorem gives $Enc^{-1}$ in Property 3.

The outsourcing side does not send the values of $x_1, x_2, \ldots, x_n$ but rather sends their encrypted values to the subcontractor and gets the computational results of $f$ back from the subcontractor, where these computational results are different from what outsourcing side wants to know. However, the outsourcing side can make $F$ from $H_1, H_2, \ldots, H_{m+1}$, in order to decrypt the results from the subcontractor to the value of $f(x_1, x_2, \ldots, x_n)$.

That is, we can realize a secret computation with its key being the set $\{H_1, H_2, \ldots, H_{m+1}\}$.

For example, if $x_1 = 0.1, x_2 = 0.5, x_3 = 0.3, f = x_1 + x_2x_3, H_1 = 1, H_2 = 2, H_3 = 3, \lambda_1 = 4.0, \lambda_2 = 5.0, \lambda_3 = 6.0$, then

$$F = \begin{bmatrix} \frac{1^0}{0!} & \frac{1^1}{1!} & \frac{1^2}{2!} \\ \frac{2^0}{0!} & \frac{2^1}{1!} & \frac{2^2}{2!} \\ \frac{3^0}{0!} & \frac{3^1}{1!} & \frac{3^2}{2!} \end{bmatrix} = \begin{bmatrix} 1 & 1 & \frac{1}{2} \\ 1 & 2 & 2 \\ 1 & 3 & \frac{9}{2} \end{bmatrix}, \quad G = \begin{bmatrix} 38.75 \\ 137.25 \\ 295.75 \end{bmatrix}.$$

We find that

$$F^{-1}G = \begin{bmatrix} 1 & 1 & \frac{1}{2} \\ 1 & 2 & 2 \\ 1 & 3 & \frac{9}{2} \end{bmatrix}^{-1} \begin{bmatrix} 38.75 \\ 137.25 \\ 295.75 \end{bmatrix}$$

$$= \begin{bmatrix} 3 & -3 & 1 \\ -2.5 & 4 & -1.5 \\ 1 & -2 & 1 \end{bmatrix} \begin{bmatrix} 38.75 \\ 137.25 \\ 295.75 \end{bmatrix} = \begin{bmatrix} 0.250 \\ 8.50 \\ 60.0 \end{bmatrix},$$

and the first entry of this result, 0.250, is equivalent to the output of $f = x_1 + x_2x_3$ with $x_1 = 0.1, x_2 = 0.5, x_3 = 0.3$.

Refs. [6][7] showed that the *IH* method is efficient under the following conditions.

Condition 1. The outsourced computation is a polynomial computation.

Condition 2. The total number of additions, subtractions, and multiplications in the computation is quite larger than the degree of the polynomial.

## IV. Application to Reliablity Enginieering

Refs. [6][7] showed that the *IH* method is very useful for computing the network reliability, which is a common topic in reliability engineering.

### *A. Computation of Network Reliability*

The network is expressed by a graph to which the following assumptions apply.

Assumption 1. Each link fails independently with a known probability, while nodes never fail.

Assumption 2. The network does not fail if and only if the source and sink are connected through links which are not in the failed state.

Section II defined the idea of a graph and related terminology; here, we define the logic underlying assumptions 1 and 2 in order to explain the topology concealment problem.

The reliability of a link is defined as the probability of a link not failing. Network reliability is defined as the probability of the network not failing. $x_i$ denotes the reliability of link $i$, in relation to computing the network reliability.

$R(G)$ denotes the network reliability of $G$. We will focus on outsourcing the computation of $R(G)$ from the reliabilities of links $x_1, x_2, \dots, x_v$.

For example, $R(G)$ in Fig. 1 is computed as $R(G) = x_1x_2 + x_3 - x_1x_2x_3$ (see refs. [11]-[13]).

The problem of computing $R(G)$ is the kernel technology used in Step 2 in the Introduction, which plays a significant role in practice. There are various models and reliability measures for networks and other systems [11]-[13], but most of them are obtained by making slight changes to the above model and $R(G)$.

However, the computation time for $R(G)$ increases exponentially as the number of links of $G$ grows. That's why reducing the burden on the reliability designer by outsourcing the work of computing $R(G)$ is so important.

Note that each $x_1, x_2, \dots, x_n$ is assumed to be close to 1, e.g., 0.9999987, 0,99997, … , because it is obvious that if the reliability of a link is not close to 1, such as 0.7, then no designer would use it.

### *B. Applying the IH Method to Network Reliability*

Refs. [6][7] pointed out the following facts.

Fact 1. $R(G)$ is expressed by a multivariable polynomial with input variables $x_1, x_2, \dots, x_v$.

Fact 2. The degree of this multivariable polynomial is $m = v$.

Fact 1 implies that Condition 1 in Subsection *B* of Section III is true when computing $R(G)$. Fact 2 implies that Condition 2 is true when computing $R(G)$ because the degree of the expression of $R(G)$ is only the number of links, whereas the computation of $R(G)$ is NP-hard [11], implying that the number of additions, subtraction, multiplications in the expression of $R(G)$ increases exponentially with the number of links. Refs. [6] showed that the *IH* method is effective for the secret computation for $R(G)$ for this reason.

### *C. Improved IH Method*

Ref. [7] found that the security strength of the *IH* method is bounded by the size of the bits which we can input in a single variable on computer. If this number becomes twice as large, then the number of bits of security of the *IH* method becomes twice as large as well. Moreover, if it becomes three times, the bits of security enlarges by three times.

Consequently, the *IH* method in ref. [6] does not have sufficient flexibility of security strength. To see this, suppose that a user requests much more bits of security, the request becomes impossible to fulfill if it goes over a threshold determined by the size of bits which can be memorized in a single variable, which is determined by the computer language and hardware specs.

Ref. [7] solved this problem by repeatedly applying the *IH* method. That is, we encrypt $x_1, x_2, \dots, x_v$ by using the *IH* method and then repeatedly encrypt the encrypted values by using the *IH* method again and again. Ref. [7] found that if we repeat the encryptions $k$ times, $k$ corresponding decryptions enable us to obtain $R(G)$. Although the computation time from the first encryption to the final decryption becomes longer, it does not cause a serious problem if $k = 2$ or 3; that is, the security strength greatly increases even when $k$ is only 2 or 3.

## V. Topology Concealment Problem

### *A. Problem*

The key issue in outsourcing the computation of $R(G)$ is concealment of the topology, not only concealment of $x_1, x_2, \dots, x_v$. Ref. [8] first pointed out this issue and gave a solution to it. Here, we will explain it in detail because ref. [8] is in Japanese.

Topology concealment problem

The secret computation is performed by not only encrypting $x_1, x_2, \dots, x_v$, but also changing the topology from $G$ (having $v$ links) to $G$', where $G$' is quite different from $G$. The outsourcing side sends the encryptions of $x_1, x_2, \dots, x_n$ and $G$' to the subcontractor, and the outsourcing side determines the value of $R(G)$ from the computational results sent back by the subcontractor and the key.

That is, the problem can be phrased as 'Can we obtain network reliability of the l. h. s. topology in Fig. 2 from the computation as for such as r. h. s. of Fig. 2?'

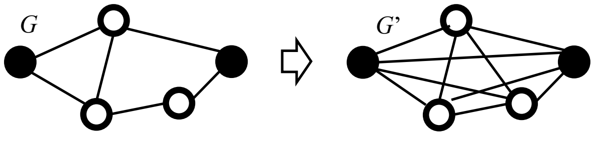

Fig. 2. Topology Concealment.

### B. Existing Approach

This subsection explains the approach to solving the topology concealment problem reported in ref. [8].

A 'contraction' is an operation on the topology that removes a link and unifies its end nodes into a single node. For example, in Fig. 3, we obtain the r. h. s. graph from l. h. s. by contracting link *e*.

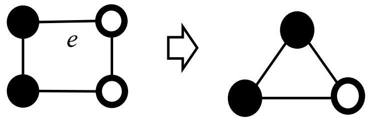

Fig. 3. Contraction.

Ref. [8] pointed out the following fact.

Fact 3. Let $G^-$ be the graph obtained by contracting link *e*. $R(G) = R(G^-)$ is true if the reliability of link *e* is 1.

Furthermore, ref. [8] defined an 'inverse contraction'. That is, if $G$ is obtained by contracting link *e* of $G^+$ , we say '$G^+$ is obtained by inverse contracting link *e* of $G$' (See Fig. 5).

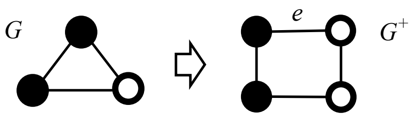

Fig. 4. Inverse contraction.

We call *e* a 'complete dummy link' and say 'add a complete dummy link' when we apply an inverse contraction.

When we want to outsource the computation of $R(G)$ but do not want to reveal the topology, one idea is to reveal $G_L^+$, which is the topology obtained by adding $L$ complete dummy links to $G$, and outsource the computation of $R(G_L^+)$ with the *IH* method. The outsourcing side can obtain $R(G)$ because Fact 3 guarantees that $R(G) = R(G_L^+)$.

Fig. 7 is obtained by adding 12 complete dummy links (the thick lines) to the graph of Fig. 5.

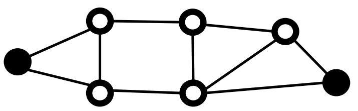

Fig. 5. Topology before applying inverse contractions.

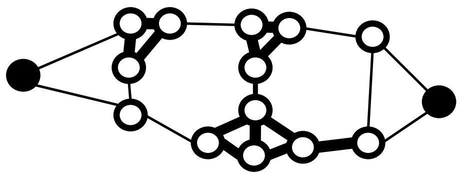

Fig. 6. Topology after adding 12 complete dummy links.

To find the original topology, a hacker must identify which links are complete dummy links. This is a very difficult problem: for example, suppose that a hacker gets the topology illustrated in Fig. 6, which is a graph with 22 links. The hacker must find the correct topology from $2^{22} = 4194304$ candidates.

Ref. [7] showed that the number of candidates is $2^{v+L}$. Accordingly, its bits of security is $\log_2(2^{v+L}) = v + L$.

### C. Problem of Adding Complete Dummy Links

We call a link with a reliability of 0 a 'zero dummy link'.

A simpler way than in ref. [8] to solve the topology concealment problem is to add a zero dummy link. Adding zero dummy links does not change the network reliability because although the added link always fails, it does not affect the network reliability. Fig. 7 shows an example of adding a zero dummy link. Note that dotted lines denote zero dummy links in the figures of this paper.

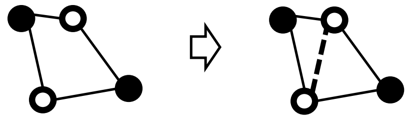

Fig. 7. Addition of a zero dummy link.

Adding complete dummy links is quite different from adding zero dummy links in terms of the transformation of the topology; adding complete dummy links increases the number of nodes while adding a zero dummy link does not change the number of nodes. If we use both kinds of dummy link, it seems that it would be more difficult for a hacker to learn the original topology, because s/he must determine not only which links are dummy links but also whether each dummy link is a complete dummy link or zero dummy link.

At this point, it is worth wondering why ref. [8] did not examine the use of zero dummy links, as their effectiveness would seem to be clear.

The reason is as follows.

1. Ref. [8] assumed that the *IH* method is used for the secret computation, because it is the only secret computation applicable to network reliability.
2. The *IH* method encrypts $x_i$ to $x_i + H_j\lambda_I$, as explained in Subsection *B* of Section III.
3. If we select $H_i$ and $\lambda_i$ uniformly at random from within an interval such as (0, 1), for example, then $1 + H_i\lambda_j$ will have almost the same probabilistic distribution as $x_i + H_i\lambda_\iota$ because $x_i$ represents the reliability of the link and every $x_i$ is close to 1, such as 0.99999987, as stated in the note at the end of Subsection *A* of Section IV.
4. Therefore, it is very difficult for a hacker to identify which links are complete dummy links by using any statistical methodology. (Ref. [8] experimentally verified this by using an appropriate statistical test methodology.)
5. That's why it is reasonable to add complete dummy links to solve the topology concealment problem.
6. However, the hacker can distinguish between $0 + H_i\lambda_j$ and $x_i + Hi\lambda_i$ under the same assumptions for $H_i$ and $\lambda_i$ in 3 because the fact that $x_i$ is close to 1 means there is an extremely high chance that $0 + H_i\lambda_j < x_i + Hi\lambda_i$
7. Accordingly, the hacker can identify which links are zero dummy links with high confidence by comparing the magnitudes of the encrypted values of the links.

The authors of Ref. [8] considered that it would be hard to make a rule to generate $H_i$ and $\lambda_i$ for the case of adding zero dummy links, but this turns out not to be the case.

## VI. Proposal

Here, we propose to add zero dummy links with the following rules for generating $H_i$ and $\lambda_i$.

RULES FOR GENERATING $H_i$ AND $\lambda_\text{i}$.

RULE 1. Every $H_j$ is generated by selecting a number uniformly at random from the interval (0, 1).
RULE 2. $\lambda_i = 1$ if link $i$ is a zero dummy link.
RULE 3. $\lambda_i = -1$ if link $i$ is not a zero dummy link.

Let $Z_1$ be the set of encrypted data obtained by applying these rules to all the links after adding $L$ dummy links.

For example, suppose we add a complete dummy link and a zero dummy link to a graph, as in Fig. 8.

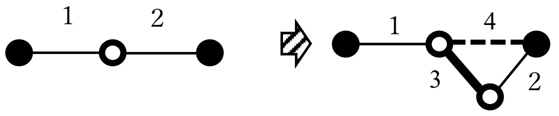

Fig. 8. Addition of dummy links.

The network reliability of the r. h. s. topology is $x_1(x_4 + x_2x_3 - x_2x_3x_4)$ with degree $m = 5$.

$Z_1$ is obtained using the above rules as follows.

$$\begin{aligned}\mathrm{Z}_1 = \{&x_1 - H_1, x_2 - H_1, 1 - H_1, 0 + H_1,\\ &x_1 - H_2, x_2 - H_2, 1 - H_2, 0 + H_2,\\ &x_1 - H_3, x_2 - H_3, 1 - H_3, 0 + H_3,\\ &x_1 - H_4, x_2 - H_4, 1 - H_4, 0 + H_4,\\ &x_1 - H_5, x_2 - H_5, 1 - H_5, 0 + H_5\}\end{aligned}$$

where $H_i$ ($i = 1, 2, 3, 4, 5$) is determined by selecting uniform random numbers from (0, 1).

If $X$ is a uniform random number in (0, 1), then $AX + B$ with constants $A$ and $B$ is a uniform random number in the interval (Min($B$, $A + B$), Max($B$, $A + B$)). Therefore, $x_1 - H_i$, $x_2 - H_i$, $1 - H_i$, $0 + H_i$ are uniform random numbers in the intervals $(x_1 - 1, x_1)$, $(x_2 - 1, x_2)$, (0, 1), (0, 1), respectively. However, $x_1$ and $x_2$ are near 1, as stated at the end of Subsection $A$ of Section IV. Therefore, it does not cause serious problem if we recognize any of $x_1 - H_i$, $x_2 - H_i$, $1 - H_i$, $0 + H_i$ as a uniform random number with interval (0, 1).

It is true that we have the possibility that $x_i - H_i$ or $x_i - H_1$ is less than 0 even though its probability is extremely small. However, this does not cause a serious problem, as $H_i$ can be regenerated again and encryptions restarted if we find an irregular encryption.

The above observation implies that a hacker will find it difficult to identify which links are dummy links even by analyzing the features of the numerical data revealed by the outsourcing side, because the numerical data are only uniform random numbers drawn from the same interval.

Regrettably, another attack can possibly identify dummy links. For example, in Fig. 8, if we sum the encrypted results of links 3 and 4, which are $1 - H_i$ and $0 + H_i$, we find the result is exactly 1. That is, if we find that the sum of the encrypted values of links $i$ and $j$ revealed by the outsourcing side is exactly 1, then these links can be identified as dummy links.

To avoid this attack, we encrypt the data of $Z_1$ again by using the original $IH$ method, not with Rules 1 - 3 to get a set of second encrypted data $Z_2$. As we repeatedly encrypt to obtain $Z_3$, $Z_4$, …, $Z_k$, the security strength increases. The procedure to obtain $\mathrm{Z}_1$ is called the first loop, and the procedures to obtain $Z_2, Z_3, \ldots, \mathrm{Z}_k$ are called the second, third, … , $k$-th loop, respectively.

This is the same scheme as in ref. [7], except that the encryptions in the first loop are executed in accordance with Rules 1 - 3.

## VII. Security Strength

There are two problems regarding the security strength. One is the security strength needed to avoid leaking information about the reliabilities of the links; the other is the security strength needed to avoid the leaking information about the topology.

However, we will only deal with the latter problem because our proposal is based on $k$ encryptions which are the same scheme as ref. [7] except for the first loop. Refs. [7] clarified that a small increase in $k$ results in a great increase in security. Accordingly, while there is an attack explained at the last of the previous section at first loop of the repetitions, it is estimated not to cause serious problem by a bit increase of $k$.

The hacker must determine whether each link is a complete dummy link, zero dummy link, or neither of them. After adding dummy links, the topology has $v + L$ links in total. Therefore, the hacker must identify the original topology from $3^{v+L}$ candidates. This implies that its security strength is $\log_2(3^{v+L}) = (v + L)\log_2(3)$ bits of security.

Subsection $B$ of Section V indicated that the security strength of the method of ref. [8] (i.e., adding only complete dummy links) is $v + L$ bits of security. Accordingly, our proposal is more secure than that one because $(v + L)\log_2(3) > v + L$.

## VIII. Numerical Test

We applied our proposal to the topology shown Fig. 9 with the reliability of every link being 0.99999.

The topology after adding 15 links is illustrated in Fig. 10. Fat links are complete dummy links and dotted links are zero dummy links. $k = 2$.

We note that this topology is simple but realistic one, because real networks for communications consist of ring structures to realize easy control of paths and ring always realizes reasonable high reliability by guaranteeing two routes between any pair of nodes on ring.

The computer environment was as follows.

| | |
|---|---|
| OS | Windows 10 home |
| CPU | Intel® Core™ I57500U CPU@ 3.40 GHz |
| RAM | 8.00GB |
| Language | C (Quadruple precision) |

The network reliability of topology in Fig. 9 obtained without encryption is 0.999999998599, while the network reliability obtained by applying our proposal to the topology in Fig. 10 is 0.999999998600. The computation time from encryption to decryption is 0.154 seconds for the former reliability value, while it is 27.4 seconds for the latter value.

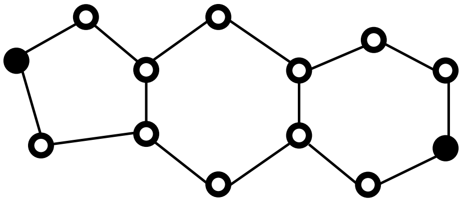

Fig. 9. Topology before Adding Dummy Links.

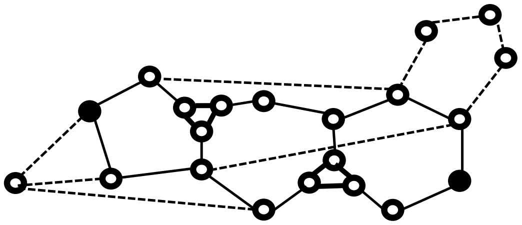

Fig 10. Topology after Adding Dummy Links.

The results revealed the following.

1. A rounding error appears but it does not cause serious errors.
2. Although computation time increases, it is not large enough to affect the designer's choice to outsource a computation if the subcontractor can use a higher-spec computer.

The bits of security required to conceal the topology is $\log_2(3^{30}) = 47.55$. This is equivalent to the difficulty of identifying the correct topology from $3^{30} = 205891132094649$ candidates.

Fig. 10 is obtained by adding 15 dummy links to Fig. 9. If we use the method of ref. [8] to add 15 complete dummy links, then its bits of security is 30, which is equivalent to the difficulty of identifying the correct topology amidst 1073741824 candidates. Thus, our proposal increases the security strength far more than the method of ref. [8], without increasing the number of dummy links.

In this test, we did not compare the method of ref. [8] and our proposal for the following reasons.

1. Adding complete dummy links increases the number of nodes while adding zero dummy links never increases it.
2. Such a difference causes different topologies after adding links
3. It is difficult or impossible to interpret the results of a comparison of such different topologies.

## IX. Conclusion

This paper proposed an improved method to solve the topology concealment problem in the secret computation of network reliability. The method adds two types of dummy link with reliabilities of one or zero, whereas the previous method adds only a single type of dummy link with a reliability of one. Our proposal prevents information about the network topology from being leaked and increases the security strength relative to that of the previous method without increasing the number of dummy links.

A numerical test showed the effectiveness of our proposal.

Future work will include conducting a more detailed analysis, improving the security strength, finding other techniques to conceal the topology, and applying our method to practical problems.

Finally, we would like to emphasize that the topology concealment problem was proposed very recently, and various similar problems will likely be found in many fields of engineering and science. The topic of this paper is one of the first works in this new field. We welcome further researches on the topology concealment problem.

## References

[1] C. Gentry, "Fully homomorphic encryption using ideal lattices," 41th ACM symp. Theory of Computing, pp. 169-178, 2009.

[2] C. Gentry and S. Halevi, "Implementing gentry's fully homomorphic encryption scheme," 11th EUROCRYPT, pp. 129-148, 2011.

[3] L. Ducas and D. Micciancio, "FHEW: Bootstrapping homomorphic encryption in less than a second", EUROCRYPT, pp. 617-640, 2015.

[4] K. Gai, M. Qiu, Y. Li, and Y. X. Liu, "Advanced fully homomorphic encryption scheme over real numbers," 4th CSCloud, pp. 64-69, 2017.

[5] I. Chillotti, N.Gama, M. Georgiava, and M. Izabachene, "TFHE: fast fully homomorphic encryption over the torus", Journal of Cryptopology, vol. 33, no. 2, pp. 34-91, 2019.

[6] T. Iseki and M. Hayashi, "Improved secure computation over real numbers and its application to reliability engineering," APNOMS 2019, INSPEC Accession Number: 19134631, 2019.

[7] N. Nakadai and M. Hayashi, "Improving the security strength of Iseki's fully homomorphic encryption," ITC-CSCC, pp. 299-304, 2020.

[8] M. Hayashi, A. Ito, and N. Nakadai, "Secret computation used in reliability engineering," REAJ, vol. 43, 2022, to appear.

[9] NIST , "Recommendation for key management – Part 1: General (Revision 4) ," NIST SP 800-57, 5.6.1, pp. 51-54, 2007.

[10] D. Kalman, "Combinatorial and functional identities in one-parameter matrices," The American Mathematical Monthly, vol. 94, no. 1, pp. 21-3, 1987.

[11] C. J. Colbourn, "The combinatorics of network reliability," New York: Oxford University Press, 1987.

[12] S. K. Chaturvedi, Network Reliability, Scrivener Publishing, 2016.

[13] L. M. Leemis, Reliability - Probabilistic models and statistical method, 2009.