Construction of Petri nets and Calculation of Elementary T-invariants for Multi-stage-Encryptions Public-Key Cryptography: MEPKC

Ryo Yamaguchi\(^1\), Qi-Wei Ge\(^2\) and Mitsuru Nakata\(^2\)

\(^1\)Graduate School of Education, Yamaguchi University, 1677-1 Yoshida, Yamaguchi-shi 753-8513, Japan
\(^2\)Faculty of Education, Yamaguchi University, 1677-1 Yoshida, Yamaguchi-shi 753-8513, Japan

E-mail: \(\{\text{k025kn, ggw, mnakata}\} @\text{yamaguchi-u.ac.jp}\)

Abstract: This paper aims at automatically generating Petri nets that are used as key generator of a public-key cryptography MEPKC. MEPKC is different from general public-key cryptography, which opens a key-generator to the public and uses the elementary T-invariants as the encryption keys to repeatedly encrypt a plain text stage by stage, so that the its security can be as strong as expected. The Petri net used as the key generator has the complicated structure that cannot calculate all elementary T-invariants, and the creator of the Petri net must be able to grasp all the elementary T-invariants. In this paper, we propose a method of generating Petri nets whose complexity is enough to be a key generator and obtaining all its the elementary T-invariants.

1. Introduction

The Internet has become so widespread that anyone can obtain and provide information easily through the open electronic network. To guarantee safety electronic communications, utilization of cryptography becomes ever important in order to avoid leak of secret information or dishonest alteration of the information [1].

There are two types of cryptography: private-key and public-key cryptosystems. Private-key cryptosystem is to use a common private key to encrypt and decrypt messages and can process encryption and decryption very fast, but it is faced with a problem how to distribute the private key through public network without leak of the secrecy [1],[2]. Public-key cryptosystem successfully solves this problem by preparing a pair of keys (public and private keys) and opening to the public key while keeping the private key secret [3].

We have proposed a new public-key cryptography MEPKC, a Petri net based Multi-stage-Encryption Public-Key Cryptography[4]. The characteristic of Petri nets that it is quite difficult to compute all the elementary T-invariants of a Petri net and nevertheless it is relatively easy to obtain one or several elementary T-invariants is well-known. Different from a usual public-key cryptography that opens a single public key to the public, our public-key cryptography opens a key generator, from which a large number of encryption keys can be generated. Using these encryption keys to repeatedly encrypt a plain text stage by stage, we can try to make the security of a cipher text be as strong as expected. We choose a Petri net possessing a large number of elementary T-invariants as the key generator and use its elementary T-invariants as the encryption keys. The Petri net used as a key generator has vast number of elementary T-invariants, and it is necessary to have the complicated structure that cannot calculate all elementary T-invariants, and the creator of the Petri net must be able to grasp all elementary T-invariants constituting private key at the same time[4]. To meet these demands, we proposed the generation method of a kind of Petri nets called \(k\)-Ring(n) and the combined Petri net \(PN_m\) having vast number of elementary T-invariants from \(k_i\)-Ring(n_i) Petri nets \(1 \leq i \leq m\)[5]. But, because the structure of the \(PN_m\) Petri net is too simple, it has not enough strength as a key generator of MEPKC. Therefore, in this paper, we propose a method of generating more complicated Petri nets by combining two \(PN_m\)s, and further discuss a method of obtaining all its elementary T-invariants.

2. Definitions of Petri Nets

In this section, we show definitions of the Petri net and some concepts of relating to it.

[Definition 1] Petri net\([6]\) is a bipartite graph, \(PN=(T, P, E, \alpha, \beta)\), where \(T, P, E, \alpha,\) and \(\beta\) are as the following.

\(T\): a set of transitions \(\{t_1, t_2, \cdots , t_{|T|}\}\)

\(P\): a set of place \(\{p_1, p_2, \cdots , p_{|P|}\}\)

\(E\): \(E^+ \cup E^–\)

\(E^+:\) a set of edges from transitions to places \(e=(t, p)\)

\(E^–:\) a set of edges from places to transitions \(e=(p, t)\)

\(\alpha\): \(\alpha(e)\) is the weights of edges \(e=(p, t)\)

\(\beta\): \(\beta(e)\) is the weights of edges \(e=(t, p)\)

[Definition 2] When there exist neither edge \((p_i, t_j)\) nor edge \((t_j, p_i)\) for any \(p_i, t_j, PN\) is called pure Petri net. The \(P \rightarrow T\) incidence matrix of a pure Petri net is expressed by \(N=N^+ − N^\rightarrow = [N^+_{pt}] − [N^\rightarrow_{pt}]\), where

\(N^+_{pt} = \begin{cases} \beta_e & \text{if } e=(t, p) \\ 0 & \text{otherwise} \end{cases}\)

\(N^\rightarrow_{pt} = \begin{cases} \alpha_e & \text{if } e=(p, t) \\ 0 & \text{otherwise} \end{cases}\)

[Definition 3] When the weights of all Petri net’s edges is 1, it is said that the Petri net is regular [7]. The regular Petri net that all transition merely has one input place and output place toward says state machine. Notations such as \(t^*, t^\rightarrow, p^*\) and \(p^\rightarrow\) mean sets of places (or transitions) shown as the following, respectively.

\(t^* = \{p \mid (p, t) \in E\}\): A set of input place of \(t\)

\(t^\rightarrow = \{p \mid (t, p) \in E\}\): A set of output place of \(t\)
\( *p = \{ t \mid (t, p) \in E \} \): A set of input transition of \( p \)
\( p^* = \{ t \mid (p, t) \in E \} \): A set of output transition of \( p \)

**Definition 4**

(1) A non-negative integer vector \( J \) satisfying \( NJ = 0 \) is called \( T \)-invariant and the set of transitions \( T_J = \{ t_i \in T | j_i \neq 0 \} \) is called support of \( J \).

(2) For a \( T \)-invariant \( J \) with support \( T_J \), if there exists no such \( T \)-invariant \( J' \) whose support \( T_{J'} \) satisfies \( T_{J'} \subset T_J \), then \( T_J \) is called minimum support.

Further, for a \( T \)-invariant \( J \) with minimum support \( T_J \), if all the values \( \{ j_i \mid t_i \in T_J \} \) have no common divisor then \( J \) is called elementary \( T \)-invariant.

\( j_i / (\sum_k j_k) \) is called canonical value of \( t_i \) in \( J \).

**Definition 5**

(1) When it is \( *p = \phi(t = \phi) \), place \( p \) (transition \( t \)) is called Source place(Source transition).

(2) When it is \( p^* = \phi(t = \phi) \), place \( p \) (transition \( t \)) is called Sink place(Sink transition).

3. On Generation of Key-Generator

In this section, we describe the proposed method of generating the \( k \)-Ring(\( n \)) Petri net and the \( PN_m \) Petri net.

The \( PN_m \) which have vast number of elementary \( T \)-invariants, is composed from \( k_i \)-Ring(\( n_i \)) Petri nets (\( 1 \leq i \leq m \)). And then, we propose a new method of generating more complicated Petri nets by combining two \( PN_m \).

The Petri net used as key generator of MEPKC should have enormous elementary \( T \)-invariants and so complicated structure that all the elementary \( T \)-invariants cannot be simply obtained. And the creator of the Petri net must be able to grasp all elementary \( T \)-invariants. A \( k \)-Ring(\( n \)) Petri net consists of \( k \) places and \( k \) transitions, and these places and transitions are connected in a ring. A \( k \)-Ring(\( n \)) Petri net is constructed by combining \( n \) \( k \)-Ring(\( 1 \)) Petri nets through sharing all the \( k \) places, and thus \( k \)-Ring(\( n \)) possesses \( k \) places and \( k \times n \) transitions. Obviously, a \( k \)-Ring(\( n \)) contains \( k^n \) elementary \( T \)-invariants. By combining plural \( k_i \)-Ring(\( n_i \)) Petri nets with different number of places, such as \( k_1 \)-Ring(\( n_1 \)), \( k_2 \)-Ring(\( n_2 \)), \( \ldots \), \( k_m \)-Ring(\( n_m \)) (\( k_1 < k_2 < \cdots < k_m \)), we get a Petri net \( PN_m \). Here the places of each \( k_i \)-Ring(\( n_i \)) are numbered as \( p_1, p_2, \ldots, p_k \), and the places numbered with the same \( p_i \) of all the \( m \) \( k_i \)-Ring(\( n_i \)) Petri nets are shared in \( PN_m \). Such a Petri net \( PN_m \) surely has enough many elementary \( T \)-invariants. Figure 1 shows the Petri net \( PN_3 \) that is composed of \( 3 \)-Ring(\( 1 \)), 5-Ring(\( 2 \)) and 7-Ring(\( 1 \)).

The \( PN_m \) is not appropriate as a key generator of MEPKC, because its structure is not complex enough. Therefore, we are to propose a new method of generating more complicated Petri nets by combining two \( PN_m \).

At first, we generate two Petri nets \( PN_{m_1} \) and \( PN_{m_2} \) by combining \( k_1 \)-Ring(\( n_1 \)), \( k_2 \)-Ring(\( n_2 \)), \( \ldots \), \( k_{m_1} \)-Ring(\( n_{m_1} \)) and \( k_1 \)-Ring(\( n_1 \)), \( k_2 \)-Ring(\( n_2 \)), \( \ldots \), \( k_{m_2} \)-Ring(\( n_{m_2} \)) respectively, where \( k_{m_1} \geq k_{m_2} \). The method of combining \( PN_{m_1} \) and \( PN_{m_2} \) is to share arbitrary place \( p_\eta \) \( (1 \leq \eta \leq k_{m_1} \) of \( PN_{m_1} \) with \( p_1 \) place of \( PN_{m_2} \), and to share \( p_{(\eta+1)} \) of \( PN_{m_1} \) with \( p_{(\eta+1)} \) \( (1 \leq \eta \leq k_{m_2}-1) \) of \( PN_{m_2} \). Therefore, all places of \( PN_{m_2} \) are shared with places of \( PN_{m_1} \). Note that, if \( p_{(\eta+1)} \) of \( PN_{m_1} \), \( p_{(\eta+1-k_{m_1})} \) of \( PN_{m_1} \) is shared with \( p_{(1+i)} \) of \( PN_{m_2} \). The Petri net combined according to the method is denoted by \( PN_{m_1+m_2} \).

Figure 2 shows the Petri net \( PN_{3+3} \) that composed 3-Ring(\( 1 \)) \times 5-Ring(\( 2 \)) \times 7-Ring(\( 1 \)) and 4-Ring(\( 1 \)) \times 5-Ring(\( 1 \)) \times 6-Ring(\( 1 \)) as \( \eta = 4 \). Note that in Figure 2, 3-Ring(\( 1 \)) \times 5-Ring(\( 2 \)) \times 7-Ring(\( 1 \)) and 4-Ring(\( 1 \)) \times 5-Ring(\( 1 \)) \times 6-Ring(\( 1 \)) are respectively indicated by solid line and broken line.

4. Enumeration of Elementary \( T \)-invariants

To enumerate all the elementary \( T \)-invariants, we first decompose \( PN_{m_1+m_2} \) into three subnets. For each subnet, we build the hierarchy graphs that are constructed only by transitions. With each of the hierarchy graphs, we enumerate all the elementary \( T \)-invariants.

4.1 Three subnets

We decompose \( PN_{m_1+m_2} \) into three subnets: \( SN_{ET_1}^{1\times 2(\eta)} \), \( SN_{ET_2}^{1\times 2(\eta)} \) and \( SN_{ET_3}^{1\times 2(\eta)} \). Let \( T_{\eta}^\prime \) be the set of transitions whose input and output places are \( p_s \) and \( p_p \) respectively.

\( SN_{ET_1}^{1\times 2(\eta)} \) is obtained by deleting place \( p_1 \) and transitions in \( \{ t \in T_{\eta}^\prime | x \neq \eta-1 \} \). \( SN_{ET_2}^{1\times 2(\eta)} \) is obtained by (i) deleting \( p_s \) and \( \{ t \in T_{\eta}^\prime | x \neq k_1 \} \); (ii) deleting \( \{ t \in T_{\eta}^\prime | x=s, s+1, \ldots, \eta-1 \} \) if \( \eta + k_2 - k_{m_1} -1 > 0 \), where \( s=\max \{ w | T_{\eta} \neq \phi, w<\eta-1 \} \); otherwise deleting
A graph is called the hierarchy graph to the elementary T-invariants of \( SN_i \) of transitions to the sink transitions. We have proved that the \( (\{ x=1 \} \cup \{ x=k_m \} ) \) \( \times \{ \eta \} \) \( T \)-invariants from \( 1 \), \( \cdots \), \( m \) are one to one corresponding to the operation for \( C \)-transitions and these rows are elementary T-invariants. Now work corresponding to transitions included in \( \tau \), where \( \tau \) is the number of all elementary T-invariants in \( ET \). Set the entries of \( L_m \) that does not.

### 4.3 Method of enumerating all the elementary T-invariants

Before enumerating elementary T-invariants, we prepare a \( R_i \times C_i \) matrix \( Mem \), where \( R_i \) is the number of all elementary T-invariants in \( ET \) \( i=1, 2, 3 \) and \( C_i \) is the number of all transitions in \( PN_1 \times 2(\eta) \).

Our process to find out all the directed paths starts from top level of each hierarchy graph. The following is the process to find out all the directed paths of \( SN_i \times 2(\eta) \).

At first, we work on \( L_1 \). Set the entries of the first \( |L_1|' \) \( + |L_1|'' \) rows of \( Mem \) corresponding to transitions included in \( L_1 \) and paste it \( |L_2| = 1 \) times, except the rows corresponding to the operation for \( L_1 \), since \( L_1 \) contains sink transitions and these rows are elementary T-invariants. Now work on \( L_2 \). Set the entries of \( |L_1|' \) \( + |L_1|'' \) \( + |L_2| [L_2] \) rows of \( Mem \) corresponding to transitions in \( L_2 \) and \( L_2 \). We copy rows corresponding to the operation for \( L_2 \), and paste it \( |L_3| - 1 \) times. Similarly, we can repeat this process till \( L_{m+1} \).

Based on the above method, we can find out all the directed paths of \( G^{\times 2(\eta)}_G \) and \( G^{\times 2(\eta)}_G \). But, we cannot apply the above method to the hierarchy graph \( G^{\times 2(\eta)}_G \) because there are two different sink resources. Therefore, we put the directed paths from the source transitions to \( p_0 \) and the directed paths from \( p_0 \) to the sink transitions together to form all the directed paths from source transitions to sink transitions. The following are some notations in the enumeration algorithm.
transitions in generally speaking, it is hard to further reduce the time complexity: in that the number of elementary \( T \)-invariants is proportional to the number of elementary \( T \)-invariants, the number of all elementary \( T \)-invariants in the entries of the rows of \( \lambda \). Thus, we obtain a \( 2(4) \times 2(4) \) matrix \( Mem \). Moreover, we have also proposed an algorithm to enumerate all elementary \( T \)-invariants of \( PN_{m+2} \) Petri net. As a future research, we need to verify whether the automatically generated \( PN_{m+2} \) Petri net has enough strength as a key generator of MEPKC.

We have proposed a method, by composing two \( PN_{m} \) Petri nets, to make the structure of the Petri net complicate enough to be used as a key generator of a public-key cryptography MEPKC. Moreover, we have also proposed an algorithm to enumerate all elementary \( T \)-invariants of \( PN_{m+2} \) Petri net. As a future research, we need to verify whether the automatically generated \( PN_{m+2} \) Petri net has enough strength as a key generator of MEPKC.

### References


