Des. Codes Cryptogr. (2017) 82:559–582
DOI 10.1007/s10623-016-0180-0

How to make a linear network code (strongly) secure

Kaoru Kurosawa[1] · **Hiroyuki Ohta**[1] · **Kenji Kakuta**[1]

Received: 6 April 2015 / Revised: 23 October 2015 / Accepted: 9 January 2016 /
Published online: 28 January 2016

Abstract A linear network code is called k-secure if it is secure even if an adversary eavesdrops at most k edges. In this paper, we show an efficient deterministic construction algorithm of a linear transformation T that transforms an (insecure) linear network code to a k-secure one for any k, and extend this algorithm to strong k-security for any k . Our algorithms run in polynomial time if k is a constant, and these time complexities are explicitly presented. We also present a concrete size of $|\mathsf{F}|$ for strong k-security, where F is the underling finite field.

Keywords Information theory · Network code · Security

Mathematics Subject Classification 94A17

1 Introduction

The notion of network code was introduced by Ahlswede et al. [1]. Li et al. [11] proved that the source node s can multicast n field elements $(m_1, \ldots, m_n)$ to a set of sink nodes $\mathsf{Sink} = \{t_1, \ldots, t_q\}$ by using a linear network code if $|\mathsf{F}| \geq |\mathsf{Sink}|$, where

$$n = \min_i \text{max-flow}(s, t_i)$$

and F is a finite field such that $m_i \in \mathsf{F}$. (Fig. 1 shows an example of a linear network code.) Jaggi et al. [8] proposed a polynomial time algorithm which can construct a linear network code from any network instance $(G(\mathcal{V}, \mathcal{E}), s, \mathsf{Sink}, n)$, where $G(\mathcal{V}, \mathcal{E})$ is the underlying network.

Communicated by M. Paterson.

A preliminary version of this paper appeared in [10].

✉ Kaoru Kurosawa
kurosawa@mx.ibaraki.ac.jp; kaoru.kurosawa.kk@vc.mx.ibaraki.ac.jp

[1] Ibaraki University, 4-12-1 Nakanarusawa, Hitachi, Ibaraki 316-8511, Japan

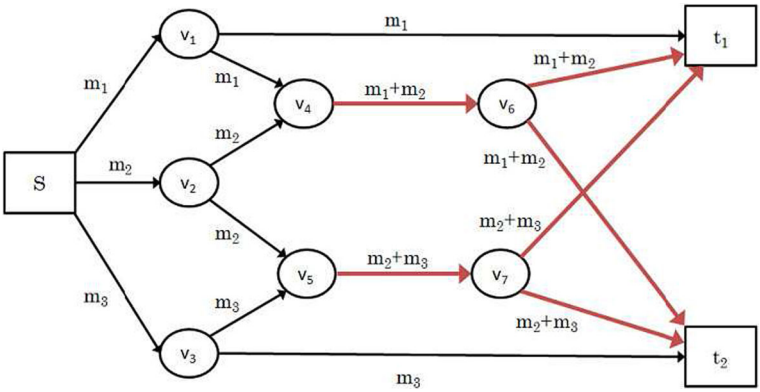

Fig. 1 Linear network coding scheme with $n = 3$

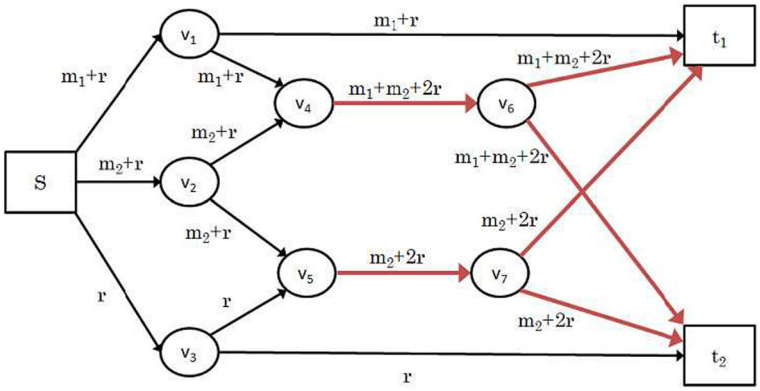

Fig. 2 1-Secure linear network coding scheme (mod 3)

Consider a model such that the source node s multicasts $(m_1, \ldots, m_{n-k}, r_1, \ldots, r_k)$ instead of $(m_1, \ldots, m_n)$, where r_i is chosen uniformly at random from the field F. We say that a linear network code is k-secure if an adversary learns no information on $(m_1, \ldots, m_{n-k})$ even by eavesdropping at most k edges.

Figure 2 shows a 1-secure linear network code. For example, $d_1 = m_1 + r$ is transmitted on the edge (s, v_1). It leaks no information on m_1 because the random element r works as one-time pad.

Cai and Yeung [4] proved that there exists a linear transformation T that makes any linear network code k-secure if $|\mathsf{F}| > \binom{|\mathcal{E}|}{k}$, where $\mathcal{E}$ is the set of edges. In fact, T is an $n \times n$ nonsingular matrix.

The advantage of this method is that it does not require changing the underlying linear network code [6]. The source node s only has to multicast $(\tilde{m}_1, \ldots, \tilde{m}_n) = (m_1, \ldots, m_{n-k}, r_1, \ldots, r_k) \times T$.

Cai and Yeung, however, only showed the *existence* of T based on a counting argument (see [4, Sect. V]). They did not show how to construct T efficiently.

Harada and Yamamoto [7] extended the notion of k-security to strong k-security. Consider any $A \subset \mathcal{E}$ and any $B \subset \{m_1, \ldots, m_n\}$ such that $|A| = |B| \leq k$. Then a linear network code is called *strongly* k-secure if A leaks no information on $\{m_1, \ldots, m_n\} \setminus B$.

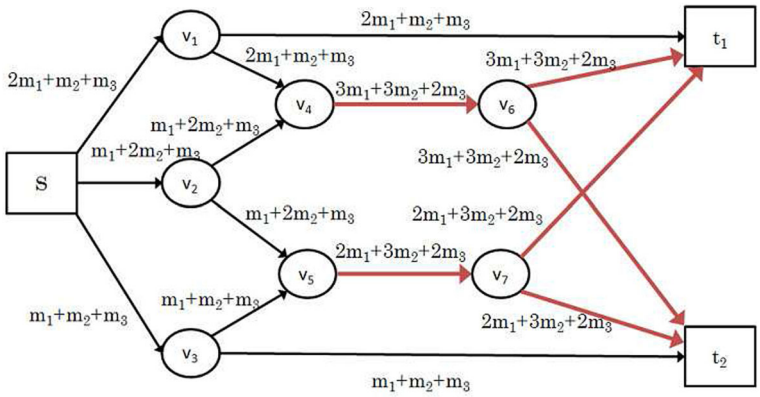

Fig. 3 Strongly 1-secure linear network coding scheme (mod 5)

Figure 3 shows a strongly 1-secure linear network code. For example, $d_1 = 2m_1+m_2+m_3$ is transmitted on the edge (s, v_1) and

- d_1 leaks no information on (m_1, m_2) since m_3 works as one-time pad.
- d_1 leaks no information on (m_2, m_3) since $2m_1$ works as one-time pad.
- d_1 leaks no information on (m_1, m_3) since m_2 works as one-time pad.

In this model, it is assumed that each m_i is independently random.

Harada and Yamamoto proved that for sufficiently large $|\mathsf{F}|$, there exists a strongly k-secure linear network code for any network instance $(G(\mathcal{V}, \mathcal{E}), s, \mathsf{Sink}, n)$ if $k < n$. However, they did not explicitly state the time complexity of their algorithm explicitly. In addition, they did not suggest a concrete size for $|\mathsf{F}|$, leaving the derivation of a sufficient condition on $|\mathsf{F}|$ as an open problem. (See "open problem" in Table 1 of [7]. They considered strong k'-security with $k' \leq k$, where ℓ is used instead of k in [7].)

In this paper, we first show an efficient deterministic construction algorithm of a linear transformation T that transforms an (insecure) linear network code to a k-secure one for any $1 \leq k < n$. We then extend this algorithm for strong k-security for any $1 \leq k < n$. Both of our algorithms run in polynomial time if k is a constant.

We explicitly present the time complexities of our algorithms. We also present a concrete size of $|\mathsf{F}|$ for strong k-security, thereby solving the open problem of Harada and Yamamoto [7].

By applying our methods to Fig. 1, we can obtain the 1-secure linear network code shown in Fig. 2, the strongly 1-secure linear network code shown in Fig. 3 and the strongly 2-secure linear network code shown in Fig. 4.

1.1 Related works

Rouayheb et al. [5] proposed a direct method to construct a k-secure linear network code from a network instance $(G(\mathcal{V}, \mathcal{E}), s, \mathsf{Sink}, n)$. This method does not use a linear transformation T, and therefore requires changes to the underlying linear network code.

Bhattad and Narayanan [2] showed how to construct weakly secure linear network codes.

Silva and Kschischang [14,15] introduced the notion of universal k-secure codes and universal strongly k-secure codes. Kurihara et al. [9] improved the universal strongly k-secure codes of [14,15]. In these schemes [9,14,15], a vector (instead of a field element) is

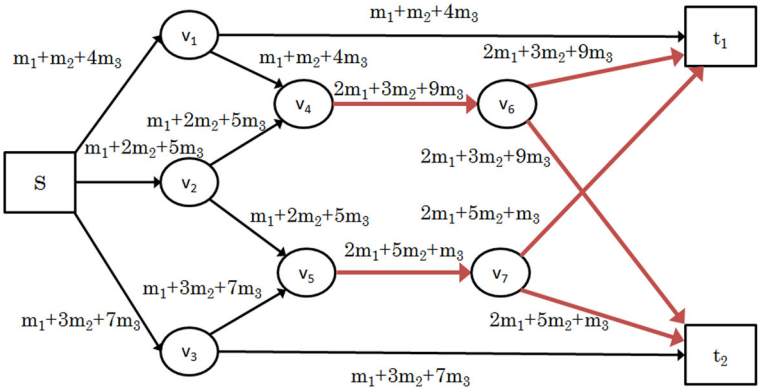

Fig. 4 Strongly 2-secure linear network coding scheme (mod 11)

transmitted over an edge. Because each element of the vector is transmitted over multiple time slots, it is assumed that the k tapped edges are fixed during the transmission period. Shioji et al. [13] considered a stronger eavesdropping model where the adversaries possess the ability to re-select the tapping edges during the transmission. They then showed that the scheme in [15] is not secure under this eavesdropping model.

Matsumoto and Hayashi [12] considered a random linear precoder at the source node and proved that it is strongly secure and universal secure if we allow arbitrary small but nonzero mutual information on the transmission symbols to the eavesdropper. In their scheme, they showed that this mutual information is upper bounded by some small quantity.

Tang et al. [16] showed a probabilistic method to construct a linear transformation T that transforms a linear network code to a k-secure one. (See "Time Complexity" of [16, p. 313].) However, they did not show the success probability and claimed (without proof) that the time complexity is $\left(\binom{|\mathcal{E}|}{k}\right)$.

2 Preliminaries

Let $H(\cdot)$ denote the Shannon entropy. For a tuple of random variables $\Lambda = (\tilde{a}_1, \ldots, \tilde{a}_\ell)$ and a subset $A = \{i_1, \ldots, i_j\} \subset \{1 \ldots \ell\}$, define

$$\Lambda_A = (\tilde{a}_{i_1}, \ldots, \tilde{a}_{i_\ell}).$$

For a vector $\boldsymbol{x} = (x_1, \ldots, x_N)$, define

$$support(\boldsymbol{x}) = \{i \mid x_i \neq 0\}.$$

Let $w_H(\boldsymbol{x})$ denote its Hamming weight and $\boldsymbol{x}^t$ denote the transpose of $\boldsymbol{x}$. For a set A, let $|A|$ denote the cardinality of A.

For an $n \times \ell$ matrix X, define X_A, $X_{A,k}$ and $X_{A,B}$ as follows.

- For $A \subset \{1, \ldots, \ell\}$, let X_A denote an $n \times |A|$ submatrix of X such that the columns are restricted to A.
- For $k < n$, let $X_{A,k}$ denote a $k \times |A|$ submatrix of X_A such that the rows are restricted to the last k rows. Namely

$$X_A = \begin{pmatrix} Y \\ X_{A,k} \end{pmatrix}$$

for some Y.
- For $B \subset \{1, \ldots, n\}$, $X_{A,B}$ denotes a $|B| \times |A|$ submatrix of X_A such that the rows are restricted to B.

Definition 1 For an $n \times \ell$ matrix X, $\mathsf{Rank}_p(X)$ denotes the set of all $A \subset \{1, \ldots \ell\}$ such that

$$|A| = rank(X_A) = p.$$

It is easy to see that the following lemma holds.

Lemma 1 *Suppose that T is a $n \times n$ nonsingular matrix T. Then* $\mathsf{Rank}_p(T \cdot X) = \mathsf{Rank}_p(X)$ *for* $p = 1, \ldots, n-1$.

$\mathcal{I}_\ell$ denotes the $\ell \times \ell$ identity matrix. F denotes a finite field and, in particular, F_p denotes a finite field of order p.

3 k-Secure linear network code

3.1 Linear network code

We define a network instance by $(G(\mathcal{V}, \mathcal{E}), s, \mathsf{Sink}, n)$:

- $G(\mathcal{V}, \mathcal{E})$ is a directed acyclic network such that each edge $e \in \mathcal{E}$ has a unit capacity, i.e., each edge can transmit one field element per time unit. G may include multiple parallel edges.
- $s \in \mathcal{V}$ is a source node.
- $\mathsf{Sink} = \{t_1, \ldots, t_q\} \subset \mathcal{V}$ is a set of sink nodes.
- n is defined as $n = \min_i \text{max-flow}(s, t_i)$, where max-flow$(s, t_i)$ denotes the maximum flow from s to $t_i \in \mathsf{Sink}$.

A linear network code for a network instance $(G(\mathcal{V}, \mathcal{E}), s, \mathsf{Sink}, n)$ is defined by an $n \times |\mathcal{E}|$ linear network coding matrix U such that

$$(m_1, \ldots, m_n) \times U = (d_1, \ldots, d_{|\mathcal{E}|}), \tag{1}$$

where $(m_1, \ldots, m_n)$ is the message that s multicasts to Sink, and d_i is the field element that is transmitted on an edge $e_i \in \mathcal{E}$. For example,

$$U = \begin{pmatrix} 100110000101100 \\ 010001100111111 \\ 001000011010011 \end{pmatrix} \tag{2}$$

is the linear network coding matrix used in Fig. 1.

Formally we say that an $n \times |\mathcal{E}|$ matrix $U = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{|\mathcal{E}|})$ over F is a linear network coding matrix for a network instance $(G(\mathcal{V}, \mathcal{E}), s, \mathsf{Sink}, n)$ if the following conditions are satisfied, where each $\boldsymbol{u}_i$ is indexed by an edge $e_i \in \mathcal{E}$.

1. If $e_i \in \mathcal{E}$ is an outgoing edge of a node $v (\neq s)$ and v has incoming edges $e_{i_1}, \ldots, e_{i_j}$, then $\boldsymbol{u}_i$ is a linear combination of $\boldsymbol{u}_{i_1}, \ldots, \boldsymbol{u}_{i_j}$.
2. Let $\{e_{i_1}, \ldots, e_{i_j}\}$ be the set of incoming edges of a sink node $t_i \in \mathsf{Sink}$. Then $Rank(\boldsymbol{u}_{i_1}, \ldots, \boldsymbol{u}_{i_j}) = n$ for each $t_i \in \mathsf{Sink}$. This condition guarantees that t_i can reconstruct $(m_1, \ldots, m_n)$.

Proposition 1 [8] *There exists a polynomial time algorithm which can construct a linear network coding matrix U from a network instance (G, s, Sink, n) if $|\mathsf{F}| \geq |\mathsf{Sink}|$.*

Proposition 2 [6, Sect. 3] *Suppose that U is a linear network coding matrix for a network instance (G, s, Sink, n). Then for any $n \times n$ nonsingular matrix T, $T \times U$ is also a linear network coding matrix for (G, s, Sink, n).*

Proposition 2 results from

$$(m_1, \ldots, m_n) \times T \cdot U = (\tilde{m}_1, \ldots, \tilde{m}_n) \times U,$$

where $(\tilde{m}_1, \ldots, \tilde{m}_n) = (m_1, \ldots, m_n) \times T$. Namely using $T \cdot U$ as a linear network coding matrix is equivalent to using U as a linear network coding matrix such that s multicasts $(\tilde{m}_1, \ldots, \tilde{m}_n)$. Each t_i can reconstruct $(m_1, \ldots, m_n)$ because T is nonsingular.

3.2 k-Secure linear network code

Consider the use of a linear network coding matrix U such that

$$(m_1, \ldots, m_{n-k}, r_1, \ldots, r_k) \times U = (d_1, \ldots, d_{|\mathcal{E}|}), \tag{3}$$

where $(m_1, \ldots, m_{n-k})$ is chosen according to some probability distribution, and each r_i is independently and uniformly chosen from F. Then, we say that U is k-secure if any k edges leak no information on $(m_1, \ldots, m_{n-k})$.

More formally, let $\tilde{M} = (\tilde{m}_1, \ldots, \tilde{m}_{n-k})$, where $\tilde{m}_i$ is the random variable induced by m_i for $i = 1, \ldots, n-k$, and let $\tilde{D} = (\tilde{d}_1, \ldots, \tilde{d}_{|\mathcal{E}|})$, where $\tilde{d}_j$ is the random variable induced by d_j for $j = 1, \ldots, |\mathcal{E}|$. Then

Definition 2 A linear network coding matrix U is k-secure if for any probability distribution on $(m_1, \ldots, m_{n-k})$, it holds that

$$H(\tilde{M} \mid \tilde{D}_A) = H(\tilde{M})$$

for any $A \subset \{1 \ldots |\mathcal{E}|\}$ such that $|A| = k$.

Proposition 3 [3, Lemma 3.1] *A linear network coding matrix $U = (u_{i,j})$ is k-secure if and only if*

$$rank(U_A) = rank(U_{A,k})$$

for any $A \subseteq \{1, \ldots, |\mathcal{E}|\}$ such that $|A| \leq k$.

In particular, U is 1-secure if and only if $u_{n,i} \neq 0$ for all i.

Cai and Yeung proved the following proposition.

Proposition 4 [4, Theorem 2] *Suppose that $|\mathsf{F}| > \binom{|\mathcal{E}|}{k}$. Then for any $n \times |\mathcal{E}|$ linear network coding matrix U, there exists an $n \times n$ nonsingular matrix T such that $V = T \times U$ is k-secure.*

The advantage of this proposition is that no changes to U are required. The source node s only has to multicast $(\tilde{m}_1, \ldots, \tilde{m}_n) = (m_1, \ldots, m_{n-k}, r_1, \ldots, r_k) \times T$. (See the paragraph at the end of Sect. 3.1.)

Cai and Yeung, however, only showed the *existence* of T based on a counting argument (see [4, Sect. V]). They did not show how to efficiently construct T.

4 Tools

4.1 Reduced coding matrix

We say that a matrix $A = (\boldsymbol{a}_1, \ldots, \boldsymbol{a}_h)$ is pairwise column independent if each pair of columns $(\boldsymbol{a}_i, \boldsymbol{a}_j)$ are linearly independent.

For an $n \times |\mathcal{E}|$ linear coding matrix U, we say that an $n \times L$ matrix $\tilde{U}$ is a reduced coding matrix of U if $\tilde{U}$ is a maximal submatrix of U such that $\tilde{U}$ is pairwise column independent. This is formally define as follows.

Definition 3 $\tilde{U} = (\tilde{\boldsymbol{u}}_1, \ldots, \tilde{\boldsymbol{u}}_L)$ is a reduced coding matrix of $U = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{|\mathcal{E}|})$ if

- $\{\tilde{\boldsymbol{u}}_1, \ldots, \tilde{\boldsymbol{u}}_L\} \subset \{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{|\mathcal{E}|}\}$,
- $\tilde{U}$ is pairwise column independent, and
- For any $\boldsymbol{u}_i$, there exists some $\tilde{\boldsymbol{u}}_j$ such that $\boldsymbol{u}_i = \beta\tilde{\boldsymbol{u}}_j$ for some $\beta \neq 0$.

We say that L is the reduced size of U.

For example, the following matrix is a reduced coding matrix of U given by Eq. (2)

$$\tilde{U} = \begin{pmatrix} 10010 \\ 01011 \\ 00101 \end{pmatrix} \tag{4}$$

and $L = 5$ is the reduced size of U.

Lemma 2 *We can compute* $(\tilde{U}, L)$ *from* $U = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{|\mathcal{E}|})$ *in time* $O(n|\mathcal{E}|^2 \cdot poly(\log|F|))$.

Proof We can check if $\boldsymbol{u}_i$ and $\boldsymbol{u}_j$ are linearly independent in $O(n \cdot poly(\log|F|))$ time. (Addition, subtraction, multiplication and devision in F takes $O(poly(\log|F|))$ times.) Therefore we can compute $(\tilde{U}, L)$ in $O(n|\mathcal{E}|^2 \cdot poly(\log|F|))$ time. □

It is easy to see that the following lemma holds.

Lemma 3 *Suppose that* T *is an* $n \times n$ *nonsingular matrix. Then,* $T \cdot \tilde{U}$ *is a reduced coding matrix of* $T \cdot U$ *if and only if* $\tilde{U}$ *is a reduced coding matrix of* U.

It is clear that the following corollaries hold from Proposition 3.

Corollary 1 *A linear network coding matrix* U *is* k*-secure if and only if*

$$rank(\tilde{U}_A) = rank(\tilde{U}_{A,k})$$

for any $A \subset \{1, \ldots, L\}$ *such that* $|A| \leq k$.

Corollary 2 *A linear network coding matrix* U *is* k*-secure if and only if for* $i = 1, \ldots, k$, $rank(\tilde{U}_{A,i}) = i$ *for any* $A_i \in \mathsf{Rank}_i(\tilde{U})$.

Corollary 3 *A linear network coding matrix* U *is* 1*-secure if and only if the last element of* $\tilde{\boldsymbol{u}}_i$ *is nonzero for each column vector* $\tilde{\boldsymbol{u}}_i$ *of* $\tilde{U}$.

4.2 How to increase Hamming weight

For two vectors $\boldsymbol{x} = (x_1, \ldots, x_N)$ and $\boldsymbol{y} = (y_1, \ldots, y_N)$, we present an algorithm $\mathsf{MaxWeight}(\boldsymbol{x}, \boldsymbol{y})$ that finds α such that

$$support(\alpha \boldsymbol{x} + \boldsymbol{y}) = support(\boldsymbol{x}) \cup support(\boldsymbol{y})$$

in $O(N)$ time. To do so, we first show an algorithm that finds $\alpha \notin S$ in $O(|S|)$ time, where $S \subset \mathsf{F}$.

Procedure: Outside(S).

Let the elements of F be $a_0, a_1, \ldots$.

1. Set $c(0) = \cdots = c(|S|) = 0$.
2. For each $a \in S$, do:
3. If $a = a_i$ for some $i \leq |S|$, then set $c(i) = 1$.
4. Let i_0 be the least i such that $c(i) = 0$. Return a_{i_0} as α.

For example,

- If $S = \{a_0, a_1, a_2\}$, then $c(0) = c(1) = c(2) = 1$ and $c(3) = 0$. Hence the above procedure returns $\alpha = a_3$.
- If $S = \{a_0, a_1, a_4\}$, then $c(0) = c(1) = 1$ and $c(2) = c(3) = 0$. Hence the above procedure returns $\alpha = a_2$.

It is easy to prove the following lemma.

Lemma 4 *If* $|\mathsf{F}| > |S|$, *then* $\mathsf{Outside}(S)$ *returns* $\alpha \in \mathsf{F}$ *such that* $\alpha \notin S$ *in* $O(|S|)$ *time.*

We present the procedure for $\mathsf{MaxWeight}(\boldsymbol{x}, \boldsymbol{y})$ as follows.

Procedure: MaxWeight $(\boldsymbol{x}, \boldsymbol{y})$.

Let $\boldsymbol{x} = (x_1, \ldots, x_N)$ and $\boldsymbol{y} = (y_1, \ldots, y_N)$.

1. Let $S_0 = \{-y_i/x_i \mid x_i \neq 0\}$.
2. $\alpha \leftarrow \mathsf{Outside}(S_0)$.
3. Return α.

Lemma 5 *For two vectors* $\boldsymbol{x} = (x_1, \ldots, x_N)$ *and* $\boldsymbol{y} = (y_1, \ldots, y_N)$, $\mathsf{MaxWeight}(\boldsymbol{x}, \boldsymbol{y})$ *returns* α *such that*

$$support(\alpha \boldsymbol{x} + \boldsymbol{y}) = support(\boldsymbol{x}) \cup support(\boldsymbol{y}) \tag{5}$$

in $O(N \cdot poly(\log |F|))$ *if* $|\mathsf{F}| > N$ *time.*

Proof Suppose that $|\mathsf{F}| > N$. Then because $|\mathsf{F}| > N \geq |S_0|$, we have $\alpha \notin S_0$ at line 2 of the above procedure according to Lemma 4. This means that $\alpha x_i + y_i \neq 0$ if $x_i \neq 0$.

Hence if $\alpha x_i + y_i = 0$, then $x_i = 0$. This means $y_i = 0$. Conversely if $x_i = y_i = 0$, then $\alpha x_i + y_i = 0$. Therefore $\alpha x_i + y_i = 0$ if and only if $x_i = y_i = 0$. In other words, $\alpha x_i + y_i \neq 0$ if and only if $x_i \neq 0$ or $y_i \neq 0$. Consequently we obtain Eq. (5).

Line 1 of the above procedure takes $O(N \cdot poly(\log |F|))$ time because computing y_i/x_i needs $O(poly(\log |F|))$ time. At line 2, $\mathsf{Outside}(S_0)$ runs in $O(N)$ time from Lemma 4. Therefore the algorithm runs in $O(N \cdot poly(\log |F|))$ time. □

For example, consider $\boldsymbol{x} = (1, 1, 1, 0)$ and $\boldsymbol{y} = (0, 4, 2, 3)$ over F_5. Then $S_0 = \{0, -4, -2\} = \{0, 1, 3\}$ at line 1. At line 2, we obtain $\alpha = 2 \notin S_0$. Then

$$\alpha \boldsymbol{x} + \boldsymbol{y} = 2 \times (1, 1, 1, 0) + (0, 4, 2, 3) = (2, 1, 4, 3).$$

Thus Eq. (5) is satisfied.

4.3 Making a nonzero row

Let X and Y be two $n \times \ell$ matrices. Let $\boldsymbol{x}_i$ denote the ith row of X and $\boldsymbol{y}_i$ denote the ith row of Y for $i = 1, \ldots, n$. For $c \in \{1, \ldots, n\}$, we write

$$Y \cong_{\text{except(c)}} X$$

if $\boldsymbol{y}_i = \boldsymbol{x}_i$ for all $i \neq c$. We also write

$$Y \cong_{\text{nonzero(c)}} X$$

if $w_H(\boldsymbol{y}_c) = \ell$ and $\boldsymbol{y}_i = \boldsymbol{x}_i$ for all $i \neq c$. We present a deterministic polynomial time algorithm that outputs an $n \times n$ nonsingular matrix T such that

$$T \cdot X \cong_{\text{nonzero(c)}} X$$

for X that does not contain a column vector $(0, \ldots, 0)^t$ and

$$T \cdot Y \cong_{\text{except(c)}} Y$$

for any Y.

Procedure: NonZeroRow(X,c)

Let $\boldsymbol{x}_i$ denote the ith row of X for $i = 1, \ldots, n$.

1. $\boldsymbol{y} :\leftarrow \boldsymbol{x}_c$.
2. $\alpha_c \leftarrow 0$.
3. For $i = 1, \ldots, n$, do:
4. If $i \neq c$, do:
5. $\alpha_i \leftarrow \mathsf{MaxWeight}(\boldsymbol{x}_i, \boldsymbol{y})$.
6. $\boldsymbol{y} \leftarrow \alpha_i \boldsymbol{x}_i + \boldsymbol{y}$.
7. Let Q be an $n \times n$ matrix such that the ith row is

$$\mathbf{q}_i = \begin{cases} (\alpha_1, \ldots, \alpha_n) & \text{if } i = c \\ (0, \ldots, 0) & \text{if } i \neq c \end{cases}$$

8. $T \leftarrow \mathcal{I}_n + Q$.
9. Return T.

Theorem 1 *Let X be an $n \times \ell$ matrix that does not contain a column vector $(0, \ldots, 0)^t$. Then the above algorithm outputs nonsingular matrix T such that*

$$T \cdot X \cong_{\text{nonzero(c)}} X$$

and

$$T \cdot Y \cong_{\text{except(c)}} Y$$

for any matrix Y in $O(n\ell \cdot poly(\log |F|))$ time if $|\mathsf{F}| > \ell$.

Proof Let $\boldsymbol{y}_i$ denote the ith row of $T \cdot X$ for $i = 1, \ldots, n$. Note that

$$T \cdot X = (\mathcal{I} + Q) \cdot X = X + Q \cdot X.$$

Therefore $\boldsymbol{y}_i = \boldsymbol{x}_i$ for $i \neq c$ and $\boldsymbol{y}_c$ is given as follows.

$$\begin{aligned} \boldsymbol{y}_c &= \boldsymbol{x}_c + \sum_{i=1}^{n} \alpha_i \boldsymbol{x}_i \\ &= (\ldots((\boldsymbol{x}_c + \alpha_1 \boldsymbol{x}_1) + \alpha_2 \boldsymbol{x}_2) + \ldots) + \alpha_n \boldsymbol{x}_n \end{aligned}$$

Suppose that $|\mathsf{F}| > \ell$. Then, MaxWeight outputs α_i correctly at line 5 according to Lemma 5. Therefore

$$w_H(\boldsymbol{y}_c) = |support(\boldsymbol{y}_c)| = |\cup_{i=1}^{n} support(\boldsymbol{x}_i)| = \ell$$

because X does not include $(0, \ldots, 0)^t$. Therefore

$$T \cdot X \cong_{\text{nonzero(c)}} X.$$

By the same argument, it is easy to see that

$$T \cdot Y \cong_{\text{except(c)}} Y$$

for any matrix Y.

Furthermore, it is clear that T is nonsingular. Finally, line 5 takes $O(\ell \cdot poly(\log |F|))$ time according to Lemma 5. Line 6 also takes $O(\ell \cdot poly(\log |F|))$ time. Hence the algorithm runs in $O(n\ell \cdot poly(\log |F|))$ time. □

5 Making a linear network coding matrix k-secure

In this section, we propose the first efficient deterministic algorithm to compute a nonsingular matrix T such that $T \times U$ is k-secure from a given linear network coding matrix U. Our algorithm runs in polynomial time if k is a constant.

Furthermore, our algorithm succeeds if $|\mathsf{F}| > \binom{L}{k}$, where L is the reduced size of U. Note that $|\mathsf{F}| > \binom{|\mathcal{E}|}{k}$ in Proposition 4 and $|\mathcal{E}| \geq L$. Therefore our sufficient condition on $|\mathsf{F}|$ is usually much smaller than that of Proposition 4.

Let $\tilde{U}$ be a reduced coding matrix of U.

5.1 Making a linear network coding matrix 1-secure

We begin by showing a polynomial time algorithm to compute an $n \times n$ nonsingular matrix T such that $T \cdot U$ is 1-secure. Our algorithm outputs T such that the last row of $T \cdot \tilde{U}$ consists of nonzero elements. Then, $T \cdot U$ is 1-secure according to Corollary 3.

Algorithm: 1-**Secure**$(\tilde{U})$.[1]

1. $T \leftarrow \mathsf{NonZeroRow}(\tilde{U}, n)$.
2. Return T.

Theorem 2 *The above algorithm outputs a nonsingular matrix T such that the last row of $T \cdot \tilde{U}$ consists of nonzero elements in $O(nL \cdot poly(\log |F|))$ time if $|\mathsf{F}| > L$, where $\tilde{U}$ is an $n \times L$ matrix.*

Proof Follows from Theorem 1. Note that no column vector of $\tilde{U}$ is $(0, \ldots, 0)^t$. □

Corollary 4 *We can construct a nonsingular matrix T such that $T \cdot U$ is 1-secure from any $n \times |\mathcal{E}|$ linear coding matrix U in $O(n|\mathcal{E}|^2 \cdot poly(\log |F|))$ time if $|\mathsf{F}| > L$, where L is the reduced size of U.*

[1] A similar idea was used in [16, p. 309]. However, they did not show how to find α that appears in MaxWeight$(\boldsymbol{x}, \boldsymbol{y})$ in polynomial time. They did not extend it to $k \geq 2$ like this paper, either.

Proof Suppose that $|\mathsf{F}| > L$ and let T be the output of 1-Secure $(\tilde{U})$. Note that $T \cdot \tilde{U}$ is a reduced coding matrix of $T \cdot U$ according to Lemma 3. Therefore $T \cdot U$ is 1-secure according to Theorem 2 and Corollary 3.

We can compute $\tilde{U}$ from $U = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{|\mathcal{E}|})$ in $O(n|\mathcal{E}|^2 \cdot poly(\log |F|))$ time according to Lemma 2. Note that $\max(n|\mathcal{E}|^2, nL) = n|\mathcal{E}|^2$. Therefore we can compute T from U in $O(n|\mathcal{E}|^2 \cdot poly(\log |F|))$ time. □

We now transform the insecure linear network code of Fig. 1 into a 1-secure one. By applying the above algorithm to $\tilde{U}$ of Eq. (4), we obtain.[2]

$$T = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix} \text{ over } \mathsf{F}_3.$$

The linear network coding matrix of Fig. 2 is $T \cdot U$, where U is given by Eq. (2). Namely Fig. 2 is 1-secure.[3]

5.2 Making a linear network coding matrix 2-secure

We now show a polynomial time algorithm to compute an $n \times n$ nonsingular matrix T such that $V = T \cdot U$ is 2-secure.

Lemma 6 *For an $n \times L$ matrix X, suppose that*

$$rank(X_{B,h-1}) = h - 1$$

for any $B \in Rank_{h-1}(X)$. Then for any $A \in \mathsf{Rank}_h(X)$, we can find a nonzero vector $\boldsymbol{a}$ of length h such that

$$X_A \cdot \boldsymbol{a} = (\boldsymbol{c}, 0^{h-1})^t$$

for some $\boldsymbol{c}$ in $O(h^3 \cdot poly(\log |F|))$ time. Furthermore,

$$rank(X_{A,h}) = h$$

if and only if the last element of $\boldsymbol{c}$ is nonzero.

Proof Suppose that $rank(X_{B,h-1}) = h - 1$ for any $B \in Rank_{h-1}(X)$. Fix $A \subset \{1, \ldots, L\}$ such that $A \in \mathsf{Rank}_h(X)$ arbitrarily. Then an $h \times h$ matrix $X_{A,h}$ includes an $(h-1) \times (h-1)$ submatrix $X_{B,h-1}$ such that $rank(X_{B,h-1}) = h - 1$. Therefore $rank(V_{A,h}) = h$ or $h - 1$.

Let

$$X_A = \begin{pmatrix} Y_1 \\ X_{A,h-1} \end{pmatrix} = \begin{pmatrix} Y_2 \\ X_{A,h} \end{pmatrix},$$

where $X_{A,h-1}$ is an $(h-1) \times h$ submatrix and $X_{A,h}$ is an $h \times h$ submatrix. For a vector $\boldsymbol{a}$ of length h, we have

$$X_A \cdot \boldsymbol{a} = \begin{pmatrix} Y_1 \cdot \boldsymbol{a} \\ X_{A,h-1} \cdot \boldsymbol{a} \end{pmatrix} = \begin{pmatrix} Y_2 \cdot \boldsymbol{a} \\ X_{A,h} \cdot \boldsymbol{a} \end{pmatrix}.$$

[2] We can often compute T with smaller $|\mathsf{F}|$ than the sufficient condition stated in Corollary 4 (see Sect. 8.)

[3] We cannot use F_2 instead of F_3 because $m_1 + m_2 + 2r = m_1 + m_2 \mod 2$. Hence Fig. 2 is not 1-secure if we use F_2.

It is easy to see that $rank(X_{A,h-1}) = h-1$ according to our assumption. Therefore we can find a nonzero vector $\boldsymbol{a}$ of length h such that

$$X_{A,h-1} \cdot \boldsymbol{a} = (0, \ldots, 0)^t$$

in $O(h^3 \cdot poly(\log |F|))$ time. We then have

$$X_A \cdot \boldsymbol{a} = (\boldsymbol{c}, 0^{h-1})^t,$$

where $\boldsymbol{c} = Y_1 \cdot \boldsymbol{a}$. Let the last element of $\boldsymbol{c}$ be c_0. Hence

$$X_{A,h} \cdot \boldsymbol{a} = (c_0, 0, \ldots, 0)^t.$$

1. If $c_0 = 0$, then it is clear that $rank(X_{A,h}) = h-1$.
2. Suppose that $rank(X_{A,h}) = h-1$. Then for some $B \subset A$ such that $|B| = h-1$, there exists a vector $\boldsymbol{b}$ of length $h-1$ such that

$$X_{B,h} \cdot \boldsymbol{b} = X_{A,h} \cdot \boldsymbol{a} = (c_0, 0, \ldots, 0)^t. \tag{6}$$

   This means that

$$X_{B,h-1} \cdot \boldsymbol{b} = (0, \ldots, 0)^t.$$

   However, $rank(X_{B,h-1}) = h-1$ according to our assumption. Hence we have $\boldsymbol{b} = (0, \ldots, 0)^t$. This means that $c_0 = 0$ according to Eq. (6).

Therefore $c_0 = 0$ if and only if $rank(X_{A,h}) = h-1$. Hence $c_0 \neq 0$ if and only if $rank(X_{A,h}) = h$. □

Algorithm: 2-**Secure**$(\tilde{U})$.

1. $T_1 \leftarrow \mathsf{NonZeroRow}(\tilde{U}, n)$.
2. $V_1 \leftarrow T_1 \cdot \tilde{U}$.
3. For each $A_i \in \mathsf{Rank}_2(\tilde{U})$, find a nonzero vector $\boldsymbol{a}_i$ of length 2 such that

$$\boldsymbol{y}_i = (V_1)_{A_i} \cdot \boldsymbol{a}_i = (\boldsymbol{c}_i, 0)^t \tag{7}$$

   for some $\boldsymbol{c}_i$.
4. Let $X = (\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots)$.
5. $T_2 \leftarrow \mathsf{NonZeroRow}(X, n-1)$.
6. $T \leftarrow T_2 \cdot T_1$.
7. Return T.

As an example, suppose that $\tilde{U}$ is a 3×2 matrix such that $rank(\tilde{U}) = 2$.

(1) At step 2, we obtain V_1 such that

$$V_1 = T_1 \cdot \tilde{U} = \begin{pmatrix} * & * \\ * & * \\ nonzero & nonzero \end{pmatrix}$$

because T_1 makes the last row of $\tilde{U}$ nonzero according to Theorem 1.

(2) At step 3, we find $(a_1, a_2) \neq (0, 0)$ such that

$$\boldsymbol{y} = V_1 \cdot \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \\ 0 \end{pmatrix}$$

for some c_1, c_2.

(3) At step 5, we compute T_2 such that

$$T_2 \cdot \boldsymbol{y} = \begin{pmatrix} c_1 \\ nonzero \\ 0 \end{pmatrix},$$

where T_2 makes the second element of $\boldsymbol{y}$ nonzero and does not change the other elements according to Theorem 1.

Now let

$$V_2 = \begin{pmatrix} v_{11} & v_{12} \\ v_{21} & v_{22} \\ v_{31} & v_{32} \end{pmatrix} = T \cdot \tilde{U} = T_2 \cdot T_1 \cdot \tilde{U}.$$

Then

(a) First we have

$$V_2 = T_2 \cdot (T_1 \cdot \tilde{U}) = T_2 \cdot \begin{pmatrix} * & * \\ * & * \\ nonzero & nonzero \end{pmatrix} = \begin{pmatrix} * & * \\ *' & *' \\ nonzero & nonzero \end{pmatrix}.$$

because T_2 does not changes the third row of $T_1 \cdot \tilde{U}$ according to Theorem 1. Hence,

$$v_{31} \neq 0 \text{ and } v_{32} \neq 0 \tag{8}$$

(b) Second we have

$$V_2 \cdot \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = T_2 \cdot V_1 \cdot \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = T_2 \cdot \boldsymbol{y} = \begin{pmatrix} c_1 \\ nonzero \\ 0 \end{pmatrix}$$

Therefore

$$rank\left(\begin{pmatrix} v_{21} & v_{22} \\ v_{31} & v_{32} \end{pmatrix}\right) = 2 \tag{9}$$

according to Lemma 6.

– Thus V_2 satisfies the condition of Corollary 2 from Eqs. (8) and (9).
– Furthermore, $V_2 = T \cdot \tilde{U}$ is a reduced coding matrix of $T \cdot U$ from Lemma 3, where U is the underlying linear network coding matrix.
– Consequently $T \cdot U$ is 2-secure from Corollary 2.

Theorem 3 *Let $\tilde{U}$ be an $n \times L$ matrix. If $|\mathsf{F}| > \binom{L}{2}$, then the above algorithm outputs a nonsingular matrix T in $O(nL^2 \cdot poly(\log |F|))$ time as follows. Let $V_2 = T \cdot \tilde{U}$. Then*

(1) *$rank((V_2)_{A,1}) = 1$ for any $A \in \mathsf{Rank}_1(V_2)$, and*
(2) *$rank((V_2)_{A,2}) = 2$ for any $A \in \mathsf{Rank}_2(V_2)$*

Proof Suppose that $|\mathsf{F}| > \binom{L}{2}$. At line 4 of 2-Secure($\tilde{U}$), the number of columns of X is equal to $\binom{L}{2}$. Therefore, according to Theorem 1, NonZeroRow outputs T_2 correctly at line 5. Namely, T_2 makes the $(n-1)$th row of $X = (\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots)$ nonzero and does not change the other rows. Also, NonZeroRow outputs T_1 correctly at line 1.

Note that $V_1 = T_1 \cdot \tilde{U}$ and

$$V_2 = T \cdot \tilde{U} = T_2 \cdot T_1 \cdot \tilde{U} = T_2 \cdot V_1.$$

Therefore

$$\mathsf{Rank}_p(V_2) = \mathsf{Rank}_p(V_1) = \mathsf{Rank}_p(\tilde{U}) \tag{10}$$

for any p according to Lemma 1. We can thus write $A \in \mathsf{Rank}_p$ instead of $A \in \mathsf{Rank}_p(\tilde{U})$.

(a) The last row of $V_1 = T_1 \cdot \tilde{U}$ consists of nonzero elements according to Theorem 1. This means that $rank((V_1)_{A,1}) = 1$ for any $A \in \mathsf{Rank}_1$.
(b) The last row of $V_2 = T_2 \cdot V_1$ also consists of nonzero elements because T_2 does not change the last row of V_1 according to Theorem 1. Therefore $rank((V_2)_{A,1}) = 1$ for any $A \in \mathsf{Rank}_1$.
(c) Consider $A_i \in \mathsf{Rank}_2$ at step 3. From Lemma 6 and (a), we can find a nonzero vector $\boldsymbol{a}_i$ that satisfies Eq. (7). Then we have

$$\begin{aligned}(V_2)_{A_i} \cdot \boldsymbol{a}_i &= (T_2 \cdot V_1)_{A_i} \cdot \boldsymbol{a}_i \\ &= T_2 \cdot ((V_1)_{A_i} \cdot \boldsymbol{a}_i) \\ &= T_2 \cdot \boldsymbol{y}_i \\ &= T_2 \cdot (\boldsymbol{c}_i, 0)^t \\ &= (\boldsymbol{c}'_i, 0)^t\end{aligned}$$

for some $\boldsymbol{c}'_i$. The last equality holds because T_2 does not change the last element of $(\boldsymbol{c}_i, 0)^t$.
(d) The last element of $\boldsymbol{c}'_i$ is nonzero because T_2 makes the $(n-1)$th element of $\boldsymbol{y}_i = (\boldsymbol{c}'_i, 0)^t$ nonzero. Thus we have $rank((V_2)_{A_i,2}) = 2$ for any $A_i \in \mathsf{Rank}_2$ from Lemma 6.

From (b) and (e), we see that (1) and (2) of this theorem hold.

It is clear that T is nonsingular. Finally the most time consuming part is line 5. The number of columns of X is $O(L^2)$. Hence line 5 runs in $O(nL^2 \cdot poly(\log |F|))$ time according to Theorem 1. Therefore the algorithm runs in $O(nL^2 \cdot poly(\log |F|))$ time. □

Corollary 5 *We can construct a nonsingular matrix T such that $T \cdot U$ is 2-secure from any $n \times |\mathcal{E}|$ linear coding matrix U in $O(n|\mathcal{E}|^2 \cdot poly(\log |F|))$ time if $|\mathsf{F}| > \binom{L}{2}$, where L is the reduced size of U.*

Proof Suppose that $|\mathsf{F}| > \binom{L}{2}$ and let T be the output of 2-Secure $(\tilde{U})$. Then $T \cdot \tilde{U}$ is a reduced coding matrix of $T \cdot U$ according to Lemma 3. Therefore, $T \cdot U$ is 2-secure according to Theorem 2 and Corollary 2.

According to Lemma 2, we can compute $\tilde{U}$ from $U = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{|\mathcal{E}|})$ in $O(n|\mathcal{E}|^2 \cdot poly(\log |F|))$ time. Note that $\max(n|\mathcal{E}|^2, nL^2) = n|\mathcal{E}|^2$. Therefore we can compute T in $O(n|\mathcal{E}|^2 \cdot poly(\log |F|))$ time from U. □

5.3 Making a linear network coding matrix k-secure

Finally we show an efficient deterministic algorithm to compute an $n \times n$ nonsingular matrix T such that $V = T \cdot U$ is k-secure for $3 \leq k < n$.

Algorithm: k-**Secure**$(\tilde{U})$.

1. $T_1 \leftarrow \mathsf{NonZeroRow}(\tilde{U}, n)$.
2. $V_1 \leftarrow T_1 \cdot \tilde{U}$.
3. For $h = 2, \ldots, k$, do:
4. For each $A_i \in \mathsf{Rank}_h(\tilde{U})$, find a nonzero vector $\boldsymbol{a}_i$ of length h such that

$$\boldsymbol{y}_i = (V_{h-1})_{A_i} \cdot \boldsymbol{a}_i = (\boldsymbol{c}_i, 0^{h-1})^t$$

for some $\boldsymbol{c}_i$.
5. Let $X = (\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots)$.
6. $T_h \leftarrow \mathsf{NonZeroRow}(X, n-h+1)$.
7. $V_h \leftarrow T_h \cdot V_{h-1}$.
8. $T \leftarrow T_k \ldots T_1$.
9. Return T.

Theorem 4 *Let $\tilde{U}$ be an $n \times L$ matrix. If $|\mathsf{F}| > \binom{L}{k}$, then the above algorithm outputs a nonsingular matrix T in $O(knL^k \cdot poly(\log |F|))$ time as follows. Let $V_k = T \cdot \tilde{U}$. Then*

$$rank((V_k)_{A,h}) = h \tag{11}$$

for any $A \in \mathsf{Rank}_h(V_k)$ for $h = 1, \ldots, k$.

Proof Suppose that $|\mathsf{F}| > \binom{L}{k}$. At line 5 of k-Secure($\tilde{U}$), the number of columns of X is at most $\binom{L}{k}$. Therefore from Theorem 1, $\mathsf{NonZeroRow}$ outputs T_h correctly at line 6. Furthermore $\mathsf{NonZeroRow}$ also outputs T_1 correctly at line 1.

We say that an $n \times L$ matrix V is bottom h full independent if $rank(V_{A,i}) = i$ for any $A \in \mathsf{Rank}_i(V)$ for $i = 1, \ldots, h$. Suppose that T is an $n \times n$ nonsingular matrix. Then $T \cdot V$ is bottom h full independent if V is bottom h full independent.

$V_1 = T_1 \cdot \tilde{U}$ is bottom 1 full independent from the property of T_1. Suppose that V_h is bottom h full independent. By applying the same argument as in the proof of Theorem 3, we can see that V_{h+1} is bottom $h+1$ full independent. Therefore V_k is bottom k full independent by induction. Hence Eq. (11) holds for any $A \in \mathsf{Rank}_h(V_k)$ for $h = 1, \ldots, k$.

It is clear that T is nonsingular. Finally the most time consuming part is line 6. The number of columns of X is $O(L^k)$. Hence line 6 runs in time $O(nL^k \cdot poly(\log |F|))$ from Theorem 1. Therefore the algorithm runs in in time $O(knL^k \cdot poly(\log |F|))$. □

Corollary 6 *We can construct a nonsingular matrix T such that $T \cdot U$ is k-secure from any $n \times |\mathcal{E}|$ linear coding matrix U in time $O((n|\mathcal{E}|^2 + knL^k) \cdot poly(\log |F|))$ if $|\mathsf{F}| > \binom{L}{k}$, where L is the reduced size of U.*

Proof Similar to the proof of Corollary 5. □

6 Strongly k-secure network coding scheme

In Eq. (1), suppose that each m_i is independently and uniformly distributed over F. Let $\tilde{m}_i$ denote this random variable induced by m_i for $i = 1, \ldots, n$, and let $\tilde{d}_j$ denote the random variable induced by d_j for $j = 1, \ldots, |\mathcal{E}|$. For $B \subset \{1, \ldots, n\}$, let $\overline{B} = \{1, \ldots, n\} \setminus B$. Then we define a strongly k-secure network coding matrix as follows.

Definition 4 A linear network coding matrix U is strongly k-secure if

$$H((\tilde{m}_1, \ldots, \tilde{m}_n)_{\overline{B}} \mid (\tilde{d}_1, \ldots, \tilde{d}_{|\mathcal{E}|})_A) = H((\tilde{m}_1, \ldots, \tilde{m}_n)_{\overline{B}}) \tag{12}$$

for any $A \subset \{1, \ldots |\mathcal{E}|\}$ and any $B \subset \{1 \ldots n\}$ such that $|A| = |B| \leq k$.

Harada and Yamamoto [7] proved the following proposition.

Proposition 5 [7, Theorem 3] *For sufficiently large $|\mathsf{F}|$, there exists a strongly k-secure linear network coding matrix for any network instance (G, s, Sink, n) if $k < n$.*

However, they did not show the time complexity of their algorithm explicitly. They did not present a concrete size of $|\mathsf{F}|$ either. Indeed, they left it as an open problem to derive a sufficient condition on $|\mathsf{F}|$. (See "open problem" in Table 1 of [7]. They considered strong k'-security with $k' \leq k$, where ℓ is used instead of k in [7].)

6.1 Necessary and sufficient condition

We can generalize Proposition 3 to strong k-security as follows.

Lemma 7 *Consider $A \subset \{1, \ldots |\mathcal{E}|\}$ and $B \subset \{1 \ldots n\}$ such that $|A| = |B|$. Then Eq. (12) holds if and only if*

$$rank(U_A) = rank(U_{A,B}). \tag{13}$$

The proof is similar to that of Proposition 3 [3, Lemma 3.1].

Proof Without loss of generality, let $A = B = \{1, \ldots, k\}$. Then we have

$$\begin{aligned}(d_1, \ldots, d_k) &= (m_1, \ldots, m_n) \cdot U_A \\ &= (m_1, \ldots, m_k) \cdot U_{A,B} + (m_{k+1}, \ldots, m_n) \cdot U_{A,\overline{B}} \\ &= (r_1, \ldots, r_k) + (s_{k+1}, \ldots, s_n)\end{aligned}$$

where

$$\begin{aligned}(r_1, \ldots, r_k) &= (m_1, \ldots, m_k) \cdot U_{A,B} \\ (s_{k+1}, \ldots, s_n) &= (m_{k+1}, \ldots, m_n) \cdot U_{A,\overline{B}}\end{aligned}$$

Let L_0 be the image space of U_A and L_1 be the image space of $U_{A,B}$.

Case 1 Suppose that $rank(U_{A,B}) = rank(U_A)$. Then $L_1 = L_0$. Since $(m_1, \ldots, m_k)$ is a random vector, $(r_1, \ldots, r_k)$ is uniformly distributed over $L_1 = L_0$. Therefore $(r_1, \ldots, r_k)$ works as one-time pad to mask $(s_{k+1}, \ldots, s_n)$. Hence Eq. (12) holds because one-time pad implies perfect secrecy.

Case 2 Suppose that $rank(U_{A,B}) < rank(U_A)$. Then $L_1 \subset L_0$ and there exists some $\boldsymbol{y} \in L \setminus L_1$. Let $\tilde{r}_i$ denote the random variable induced by r_i for $i = 1, \ldots, k$, and $\tilde{s}_i$ denote the random variable induced by s_i for $i = k+1, \ldots, n$. Then we have

$$\begin{aligned}&\Pr((\tilde{d}_1, \ldots, \tilde{d}_k) = \boldsymbol{y} \mid (\tilde{s}_1, \ldots, \tilde{s}_k) = (0, \ldots, 0)) \\ &= \Pr((\tilde{r}_1, \ldots, \tilde{r}_k) = \boldsymbol{y}) \\ &= 0\end{aligned}$$

On the other hand, $\Pr((\tilde{d}_1, \ldots, \tilde{d}_k) = \boldsymbol{y}) > 0$ because each m_i is independently random. Therefore

$$\Pr((\tilde{d}_1, \ldots, \tilde{d}_k) = \boldsymbol{y} \mid (\tilde{s}_1, \ldots, \tilde{s}_k) = (0, \ldots, 0)) \neq \Pr((\tilde{d}_1, \ldots, \tilde{d}_k) = \boldsymbol{y})$$

Hence $(\tilde{d}_1, \ldots, \tilde{d}_k)$ and $(\tilde{s}_1, \ldots, \tilde{s}_k)$ are not independent. This means that Eq. (12) does not hold.

□

Theorem 5 *A linear network coding matrix $U = (u_{i,j})$ is strongly k-secure if and only if*

$$rank(U_A) = rank(U_{A,B})$$

for any $A \subset \{1, \ldots, |\mathcal{E}|\}$ and any $B \subset \{1, \ldots, n\}$ such that $|A| = |B| \leq k$.

Proof From Lemma 7. □

6.2 Relation to reduced coding matrix

Corollary 7 *A linear network coding matrix U is strongly 1-secure if and only if each element of $\tilde{U}$ is nonzero, where $\tilde{U}$ be a reduced coding matrix of U.*

Corollary 8 *A linear network coding matrix U is strongly k-secure if and only if*

$$rank(\tilde{U}_A) = rank(\tilde{U}_{A,B})$$

for any $A \in \mathsf{Rank}_p(\tilde{U})$ and any $B \subset \{1, \ldots, n\}$ such that $|B| = p$ for $p = 1, \ldots, k$, where $\tilde{U}$ be a reduced coding matrix of U.

7 How to make a linear network coding matrix strongly 1-secure

In this section, we show the first efficient deterministic algorithm which computes a non-singular matrix T such that $T \times U$ is strongly k-secure from a given linear network coding matrix U. In particular, it runs in polynomial time if k is a constant.

Let $\tilde{U}$ be a reduced coding matrix of U and let L be the reduced size of U. Then our algorithm succeeds if

$$|\mathsf{F}| > L + \sum_{i=1}^{k-1} \binom{n-1}{i} \binom{L}{i+1}.$$

7.1 How to make a linear network coding matrix strongly 1-secure

We first show a polynomial time algorithm which computes an $n \times n$ nonsingular matrix T such that $T \cdot U$ is strongly 1-secure. In fact, our algorithm outputs T such that each element of $T \cdot \tilde{U}$ is nonzero. Then $T \cdot U$ is strongly 1-secure from Corollary 7.

Algorithm: Strongly 1-Secure$(\tilde{U})$.

1. $T_0 \leftarrow \mathsf{NonZeroRow}(\tilde{U}, n)$.
2. $V \leftarrow T_0 \cdot \tilde{U}$.
   Let $\boldsymbol{x}_i$ be the ith row of V.
3. For $i = 1, \ldots, n-1$, do:
4. $\quad \beta_i \leftarrow \mathsf{MaxWeight}(\boldsymbol{x}_n, \boldsymbol{x}_i)$.
5. Let

$$T_1 \leftarrow \begin{pmatrix} & & \beta_1 \\ & \mathcal{I}_{n-1} & \vdots \\ & & \beta_{n-1} \\ 0 & \cdots \quad 0 & 1 \end{pmatrix}$$

6. Return $T = T_1 \times T_0$.

Theorem 6 *Let $\tilde{U}$ be an $n \times L$ matrix. Then the above algorithm outputs a nonsingular matrix T such that each element of $T \cdot \tilde{U}$ is nonzero in time $O(nL \cdot poly(\log |F|))$ if $|\mathsf{F}| > L$.*

Proof Note that

$$T \cdot \tilde{U} = T_1 \cdot T_0 \cdot \tilde{U} = T_1 \cdot V = \begin{pmatrix} \beta_1 \boldsymbol{x}_n + x_1 \\ \vdots \\ \beta_{n-1} \boldsymbol{x}_n + x_{n-1} \\ \boldsymbol{x}_n \end{pmatrix}$$

Suppose that $|\mathsf{F}| > L$. Then from Theorem 1, we have $w_H(\boldsymbol{x}_n) = L$ at line 2. Then from Lemma 5, we have $w_H(\beta_i \boldsymbol{x}_n + \boldsymbol{x}_i) = L$ for $i = 1, \ldots, n-1$. Therefore each element of $T \cdot \tilde{U}$ is nonzero.

Further it is clear that T is nonsingular. Finally, line 1 takes time $O(nL \cdot poly(\log |F|))$ from Theorem 2, line 2 takes time $O(nL \cdot poly(\log |F|))$, line 4 takes time $O(L \cdot poly(\log |F|))$ from Lemma 5 and line 6 takes time $O(n^2 \cdot poly(\log |F|))$. Hence the algorithm runs in time $O(nL \cdot poly(\log |F|))$. □

Corollary 9 *We can construct a nonsingular matrix T such that $T \cdot U$ is strongly 1-secure from any $n \times |\mathcal{E}|$ linear coding matrix U in time $O(n|\mathcal{E}|^2 \cdot poly(\log |F|))$ if $|\mathsf{F}| > L$, where L is the reduced size of U.*

Proof Similar to the proof of Corollary 5 where we use Corollary 7. □

We transform the insecure linear network code of Fig. 1 into a strongly 1-secure one. By applying the above algorithm to $\tilde{U}$ of Eq. (4), we obtain[4]

$$T = \begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{pmatrix} \text{ over } \mathsf{F}_5.$$

The linear network coding matrix of Fig. 3 is $T \cdot U$, where U is given by Eq. (2). Namely Fig. 3 is strongly 1-secure.[5]

7.2 How to make a linear network coding matrix strongly 2-secure

We next show a polynomial time algorithm which computes an $n \times n$ nonsingular matrix T such that $V = T \cdot U$ is strongly 2-secure.

Definition 5 For a subset $D \subset \{1, \ldots, n\}$ and an $n \times \ell$ matrix E, we say that a vector $\boldsymbol{y} = (y_1, \ldots, y_n)^t$ is a D-zero image of E if (1) $\boldsymbol{y} = E \cdot \boldsymbol{a}$ for some nonzero vector $\boldsymbol{a}$ and (2) $y_i = 0$ for each $i \in D$.

For example,

$$\boldsymbol{y} = E \cdot \boldsymbol{a} = (0, y_2, \ldots, y_n)^t$$

is a $\{1\}$-zero image of E.

Lemma 8 *For a $p \times p$ matrix X, suppose that*

$$rank(X_{B,\{1,\ldots,p-1\}}) = p - 1$$

[4] We can often compute T with smaller $|\mathsf{F}|$ than the sufficient condition stated in Corollary 9. See Sect. 8.

[5] We cannot use F_3 instead of F_5 because $3m_1 + 3m_2 + 2m_3 = m_3 \mod 3$. Hence Fig. 3 is not strongly 1-secure if we use F_3.

for any $B \in Rank_{p-1}(X)$. *Then we can find a nonzero vector* $\boldsymbol{a}$ *of length* p *such that*

$$X \cdot \boldsymbol{a} = (0^{p-1}, c)^t$$

for some c *in time* $O(p^3 \cdot poly(\log |F|))$. *Further*

$$rank(X) = p$$

if and only if c *is nonzero.*

Proof Similar to the proof of Lemma 6. □

Algorithm: Strongly 2-**Secure**$(\tilde{U})$.

1. $T_0 \leftarrow \mathsf{NonZeroRow}(\tilde{U}, 1)$.
2. $V_0 \leftarrow T_0 \cdot \tilde{U}$.
3. For $h = 1, \ldots, n-1$, do:
4. For each $A_i \in \mathsf{Rank}_2(\tilde{U})$, find a $\{h\}$-zero image $\boldsymbol{y}_i$ of $(V_{h-1})_{A_i}$.
5. $W_h \leftarrow (\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots)$.
6. $T_h \leftarrow \mathsf{NonZeroRow}((V_{h-1}, W_1, \ldots, W_h), h+1)$.
7. $V_h \leftarrow T_h \cdot V_{h-1}$.
8. $W_1 \leftarrow T_h \cdot W_1$.
9. $\vdots$
10. $W_h \leftarrow T_h \cdot W_h$.
11. $T \leftarrow T_{n-1} \cdot \ldots \cdot T_0$.
12. Return T.

We illustrate how the algorithm proceeds for $n = 3$.

1. At line 2:

$V_0 = T_0\tilde{U}$
non-zero
***
***

2. In the first loop, at line 5,

$V_0 = T_0\tilde{U}$	W_1
non-zero	$0, \ldots, 0$
***	***
***	***

3. In the first loop, after line 8,

$V_1 = T_1T_0\tilde{U}$	W_1
non-zero	$0, \ldots, 0$
non-zero	non-zero
***	***

4. In the second loop, at line 5,

$V_1 = T_1T_0\tilde{U}$	W_1	W_2
non-zero	$0, \ldots, 0$	***
non-zero	non-zero	$0, \ldots, 0$
***	***	***

5. In the second loop, after line 10,

$V_2 = T_2T_1T_0\tilde{U}$	W_1	W_2
non-zero	$0, \ldots, 0$	***
non-zero	non-zero	$0, \ldots, 0$
non-zero	non-zero	non-zero

We can see the following from the last table.

- Each element of V_2 is nonzero.
- From Lemma 8, for any $A_i \in \mathsf{Rank}_2(V_2)$,
  - $rank((V_2)_{A_i,B}) = 2$ for $B = \{1, 2\}$ and $B = \{1, 3\}$.
  - $rank((V_2)_{A_i,B}) = 2$ for $B = \{2, 3\}$.

Hence $V = (T_2T_1T_0)U$ is strongly 2-secure from Corollarys 7 and 8.

Theorem 7 *Let $\tilde{U}$ be an $n \times L$ matrix. Define*

$$\lambda = L + (n-1)\binom{L}{2}.$$

If $|\mathsf{F}| > \lambda$. then the above algorithm outputs an $n \times n$ nonsingular matrix T in time $O(n^2\lambda \cdot poly(\log |F|))$ such as flows. Let $V_{n-1} = T \cdot \tilde{U}$. Then

(1) *each element of V_{n-1} is nonzero and*
(2)

$$rank((V_{n-1})_{A,B}) = 2 \tag{14}$$

for any $A \in \mathsf{Rank}_2(V_{n-1})$ and any $B \subset \{1, \ldots, n\}$ such that $|B| = 2$.

Proof Suppose that $|\mathsf{F}| > \lambda$. At line 6, the number of columns of $(V_{h-1}, W_1, \ldots, W_h)$ is at most λ. Hence NonZeroRow outputs T_h correctly from Theorem 1. Namely T_h makes the $(h+1)$th row of $(V_{h-1}, W_1, \ldots, W_h)$ nonzero and does not change the other rows. Also NonZeroRow outputs T_0 correctly at line 1, too.

For $1 \le j \le n-1$, it holds that

$$\mathsf{Rank}_2(V_j) = \mathsf{Rank}_2(\tilde{U}) \tag{15}$$

from Lemma 1 because $V_j = T_j \ldots T_0 \cdot \tilde{U}$. In this sense, we write $A \in \mathsf{Rank}_2$ instead of $A \in \mathsf{Rank}_2(\tilde{U})$.

(a) The first row of $V_0 = T_0 \cdot \tilde{U}$ consists of nonzero elements from Theorem 1.
(b) Suppose that

- the first h rows of V_{h-1} consists of nonzero elements, and
- $rank((V_{h-1})_{A,B}) = 2$ for any $A \in \mathsf{Rank}_2$ and any $B \subset \{1, \ldots, h\}$ such that $|B| = 2$.

Then we will show that

- the first $h+1$ rows of V_h consists of nonzero elements, and
- $rank((V_h)_{A,B}) = 2$ for any $A \in \mathsf{Rank}_2$ and any $B \subset \{1, \ldots, h+1\}$ such that $|B| = 2$.

(c) T_h makes the $(h+1)$th row of $(V_{h-1}, W_1, \ldots, W_h)$ nonzero and does not change the other rows. Therefore the first $h+1$ rows of $V_h = T_h \cdot V_{h-1}$ consists of nonzero elements from our assumption.

Since T_h does not change the first h rows, we have $rank((V_h)_{A,B}) = 2$ for any $A \in \mathsf{Rank}_2$ and any $B \subset \{1, \ldots, h\}$ such that $|B| = 2$ from our assumption.

Now consider $B = \{j, h+1\}$ such that $j \in \{1, \ldots, h\}$.

– Each column vector $\boldsymbol{y}_i$ of W_h is a $\{h\}$-image of $(V_{h-1})_{A_i}$, where $A_i \in \mathsf{Rank}_2$. Therefore

$$\boldsymbol{y}_i = (\boldsymbol{y}_{i,1}0, y_{h+1,i}, \boldsymbol{y}_{i,2})^t = (V_{h-1})_{A_i} \cdot \boldsymbol{a}_i,$$

for some nonzero vector $\boldsymbol{a}_i$, where

$$\begin{aligned} \boldsymbol{y}_{i,1} &= (y_{1,i}, \ldots, y_{h-1,i}) \\ \boldsymbol{y}_{i,2} &= (y_{h+2,i}, \ldots, y_{n,i}) \end{aligned}$$

Multiply T_h to the both hand sides. Then we have

$$\begin{aligned} T_h \cdot \boldsymbol{y}_i &= (\boldsymbol{y}_{i,1}0, \text{nonzero}, \boldsymbol{y}_{i,2})^t \\ &= (T_h \cdot V_{h-1})_{A_i} \cdot \boldsymbol{a}_i \\ &= (V_h)_{A_i} \cdot \boldsymbol{a}_i \end{aligned}$$

because T_h only makes the $(h+1)$th element of $\boldsymbol{y}_i$ nonzero. Therefore we have

$$(V_h)_{A_i} \cdot \boldsymbol{a}_i = (\boldsymbol{y}_{i,1}0, \text{nonzero}, \boldsymbol{y}_{i,2})^t.$$

Then from Lemma 8, we have $rank((V_h)_{A_i,B}) = 2$ for $B = \{h, h+1\}$ and $A_i \in \mathsf{Rank}_2$.

– By applying the same argument to $W_1, \ldots, W_{h-1}$, we can see that $rank((V_h)_{A,B}) = 2$ for any $A \in \mathsf{Rank}_2$ and $B = \{j, h+1\}$ such that $j \in \{1, \ldots, h-1\}$.

Therefore $rank((V_h)_{A,B}) = 2$ for any $A \in \mathsf{Rank}_2$ and any $B \subset \{1, \ldots, h+1\}$ such that $|B| = 2$.

Consequently (1) and (2) of this theorem hold by induction.

It is clear that T is nonsingular. Finally, the most time consuming part is line 6. It takes time $O(n\lambda \cdot poly(\log |F|))$ from Theorem 1. Hence the algorithm runs in time $O(n^2\lambda \cdot poly(\log |F|))$. □

Corollary 10 *We can construct a nonsingular matrix T such that $T \cdot U$ is strongly 2-secure from any $n \times |\mathcal{E}|$ linear coding matrix U in time $O((n|\mathcal{E}|^2 + n^2\lambda) \cdot poly(\log |F|))$ if $|\mathsf{F}| > \lambda$, where*

$$\lambda = L + (n-1)\binom{L}{2}$$

and L is the reduced size of U.

Proof Similar to the proof of Corollary 5, where we use Corollary 8. □

We transform the insecure linear network code of Fig. 1 into a strongly 2-secure one. By applying the above algorithm to $\tilde{U}$ of Eq. (4), we obtain[6]

$$T = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \\ 4 & 5 & 7 \end{pmatrix} \text{ over } \mathsf{F}_{11}.$$

The linear network coding matrix of Fig. 4 is $T \cdot U$, where U is given by Eq. (2). Namely Fig. 4 is strongly 2-secure[7]

[6] We can often compute T with smaller $|\mathsf{F}|$ than the sufficient condition stated in Corollary 10. See Sect. 8.

[7] We cannot use F_7 instead of F_{11} because $m_1 + 3m_2 + 7m_3 = m_1 + 3m_2 \mod 7$. Hence Fig. 4 is not strongly 2-secure if we use F_7.

7.3 How to make a linear network coding matrix strongly k-secure

We finally show an efficient algorithm which computes an $n \times n$ nonsingular matrix T such that $V = T \cdot U$ is strongly k-secure for $3 \leq k < n$.

Algorithm: Strongly k-Secure $(\tilde{U})$.

1. $T_0 \leftarrow \mathsf{NonZeroRow}(\tilde{U}, 1)$.
2. $V_0 \leftarrow T_0 \cdot \tilde{U}$ and $X_0 \leftarrow V_0$.
3. For $h = 1, \ldots, n-1$, do:
4. For each $D_j \subset \{1, \ldots, h\}$ such that $h \in D_j$ and $|D_j| < k$, do:
5. For each $A_i \in \mathsf{Rank}_{|D_j|+1}(\tilde{U})$, find a D_j-zero image $\boldsymbol{y}_i$ of $(V_{h-1})_{A_i}$.
6. $W_{D_j} \leftarrow (\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots)$.
7. $T_h \leftarrow \mathsf{NonZeroRow}((V_{h-1}, X_{h-1}, W_{D_1}, W_{D_2}, \ldots), h+1)$.
8. $V_h \leftarrow T_h \cdot V_{h-1}$.
9. $X_h \leftarrow T_h \cdot (X_{h-1}, W_{D_1}, W_{D_2}, \ldots)$.
10. $T \leftarrow T_{n-1} \cdot \ldots \cdot T_0$.
11. Return T.

We illustrate how the algorithm proceeds for $k = 3$ and $n = 4$.

1. At line 2,

$V_0 = T_0\tilde{U}$
non-zero
***
***
***

2. At the end of the 1st loop,

$V_1 = T_1T_0\tilde{U}$	$W_{\{1\}}$
non-zero	$0, \ldots, 0$
non-zero	non-zero
***	***
***	***

3. At the end of the 2nd loop,

$V_2 = T_2T_1T_0\tilde{U}$	$W_{\{1\}}$	$W_{\{2\}}$	$W_{\{1,2\}}$
non-zero	$0, \ldots, 0$	***	$0, \ldots, 0$
non-zero	non-zero	$0, \ldots, 0$	$0, \ldots, 0$
non-zero	non-zero	non-zero	non-zero
***	***	***	***

4. At the end of the 3rd loop,

$V_3 = T_3 \ldots T_0\tilde{U}$	$W_{\{1\}}$	$W_{\{2\}}$	$W_{\{1,2\}}$	$W_{\{3\}}$	$W_{\{1,3\}}$	$W_{\{2,3\}}$
non-zero	$0, \ldots, 0$	***	$0, \ldots, 0$	***	$0 \ldots 0$	***
non-zero	non-zero	$0, \ldots, 0$	$0, \ldots, 0$	***	***	$0 \ldots 0$
non-zero	non-zero	non-zero	non-zero	$0 \ldots 0$	$0 \ldots 0$	$0 \ldots 0$
non-zero	non-zero	non-zero	non-zero	non-zero	non-zero	non-zero

We can see the following from the last table.

– Each element of V_3 is nonzero.
– By applying Lemma 8 to the columns indexed by $W_{\{1\}}$, $W_{\{2\}}$ and $W_{\{3\}}$, we can see that $rank((V_3)_{A_i,B}) = 2$ for any $A_i \in \mathsf{Rank}_2(V_3)$ and for any $B \subset \{1, 2, 3, 4\}$ with $|B| = 2$.
– By applying Lemma 8 to the columns indexed by $W_{\{1,2\}}$, $W_{\{1,3\}}$ and $W_{\{2,3\}}$, we can see that $rank((V_3)_{A_i,B}) = 3$ for any $A_i \in \mathsf{Rank}_3(V_3)$ and for any $B \subset \{1, 2, 3, 4\}$ with $|B| = 3$.

Therefore$V = (T_3T_2T_1T_0)U$ is strongly 3-secure from Corollarys 7 and 8.

Theorem 8 *Let $\tilde{U}$ be an $n \times L$ matrix. Define*

$$\lambda = L + \sum_{i=1}^{k-1} \binom{n-1}{i}\binom{L}{i+1}.$$

If $|\mathsf{F}| > \lambda$, Then the above algorithm outputs an $n \times n$ nonsingular matrix T in time $O(n^2\lambda \cdot poly(\log|F|))$ such that

$$rank((T \cdot U)_{A,B}) = p \tag{16}$$

for any $A \in \mathsf{Rank}_p(\tilde{U})$ and any $B \subset \{1, \ldots, n\}$ with $|B| = p$ for $p = 1, \ldots, k$.

Proof Suppose that $|\mathsf{F}| > \lambda$. At line 7, the number of columns of the matrix which is the input to NonZeroRow is at most λ. Hence NonZeroRow outputs T_h correctly from Theorem 1. Further NonZeroRow outputs T_0 correctly at line 1, too.

We say that an $n \times L$ matrix V is top (h, k) full independent if $rank(V_{A,B}) = |B|$ for any $B \subset \{1, \ldots, h\}$ such that $|B| \leq k$ and any $A \in \mathsf{Rank}_{|B|}(V)$. Suppose that T is an $n \times n$ nonsingular matrix. Then $T \cdot V$ is top (h, k) full independent if V is top (h, k) full independent.

$V_0 = T_0 \cdot \tilde{U}$ is top $(1, 1)$ full independent because the first row of V_0 consists of nonzero elements from the property of T_0. Suppose that V_{h-1} is top (h, k) full independent. By applying the same argument as in the proof of Theorem 7, we can see that V_h is top $(h+1, k)$ full independent. Therefore V_{n-1} is top (n, k) full independent by induction. Hence Eq. (16) holds for any $A \in \mathsf{Rank}_p(\tilde{U})$ and any $B \subset \{1, \ldots, n\}$ with $|B| = p$ for $p = 1, \ldots, k$.

Furthermore, it is clear that T is nonsingular. Finally, the most time consuming part is line 7. It takes $O(n\lambda \cdot poly(\log|F|))$ time according to Theorem 1. Hence, the algorithm runs in $O(n^2\lambda \cdot poly(\log|F|))$ time. □

Corollary 11 *We can construct a nonsingular matrix T from any $n \times |\mathcal{E}|$ linear coding matrix U such that $T \cdot U$ is strongly k-secure in $O((n|\mathcal{E}|^2 + n^2\lambda) \cdot poly(\log|F|))$ time if $|\mathsf{F}| > \lambda$, where*

$$\lambda = L + \sum_{i=1}^{k-1} \binom{n-1}{i}\binom{L}{i+1}.$$

Proof Similar to the proof of Corollary 5, where we use Corollary 8. □

8 Summary

We have proposed an efficient deterministic construction algorithm of a linear transformation T that transforms a linear network code to a k-secure one for any $1 \leq k < n$. We have also extended this algorithm to strong k-security for any $1 \leq k < n$. Our algorithms run in

polynomial time if k is a constant, and these time complexities are explicitly presented. We also have presented a concrete size of $|\mathsf{F}|$ for strong k-security,

The condition on $|\mathsf{F}|$ in Lemma 5 is a sufficient condition. Therefore $\mathsf{MaxWeight}(\boldsymbol{x}, \boldsymbol{y})$ succeeds with smaller $|\mathsf{F}|$ as long as $|\mathsf{F}| > |S_0|$. For example, if the Hamming weight of $\boldsymbol{y}$ is small, then $|S_0|$ is small. Hence $|\mathsf{F}|$ can be small. For the same reason, our construction algorithms for the transformation matrix T may succeed with smaller $|\mathsf{F}|$ than the sufficient condition on $|\mathsf{F}|$ that is stated in each theorem.

Further work should explore if there exists a network instance such that our sufficient condition on $|\mathsf{F}|$ is tight.

References

1. Ahlswede R., Cai N., Li S.-Y.R., Yeung R.W.: Network information flow. IEEE Trans. Inf. Theory **46**(4), 1204–1216 (2000).
2. Bhattad K., Narayanan K.R.: Weakly secure network coding. In: Proceedings of NetCod 2005, April (2005).
3. Cai N., Yeung R.W.: A security condition for multi-source linear network coding. In: Proceedings of IEEE ISIT, 24–29 June, pp. 561–565 (2007).
4. Cai N., Yeung R.W.: Secure network coding on a wiretap network. IEEE Trans. Inf. Theory **57**(1), 424–435 (2011).
5. El Rouayheb S., Soljanin E., Sprintson A.: Secure network coding for wiretap networks of type II. IEEE Trans. Inf. Theory **58**(3), 1361–1371 (2012).
6. Feldman J., Malkin T., Servedio R., Stein C.: On the capacity of secure network coding. In: 42nd Annual Allerton Conference on Communication, Control, and Computing (2004).
7. Harada K., Yamamoto H.: Strongly secure linear network coding. IEICE Trans. **91–A**(10), 2720–2728 (2008).
8. Jaggi S., Sanders P., Chou P.A., Effros M., Egner S., Jain K., Tolhuizen L.M.G.M.: Polynomial time algorithms for multicast network code construction. IEEE Trans. Inf. Theory **51**(6), 1973–1982 (2005).
9. Kurihara J., Uematsu T., Matsumoto R.: Explicit construction of universal strongly secure network coding via MRD codes. In: Proceedings of IEEE ISIT 2012, Cambridge, MA, pp. 1488–1492 (2012).
10. Kurosawa K., Ohta H., Kakuta K.: How to construct strongly secure network coding scheme. In: ICITS, pp. 1–17 (2013).
11. Li S.Y.R., Yeung R.W., Cai N.: Linear network coding. IEEE Trans. Inf. Theory **49**(2), 371–381 (2003).
12. Matsumoto R., Hayashi M.: Universal Strongly Secure Network Coding with Dependent and Non-uniform Messages. (2012). arXiv:1111.4174.
13. Shioji E., Matsumoto R., Uyematsu T.: Vulnerability of MRD-code-based universal secure network coding against stronger eavesdroppers. IEICE Trans. **93-A**(11), 2026–2033 (2010).
14. Silva D., Kschischang F.R.: Universal weakly secure network coding. In: Proceedings of IEEE ITW 2009, pp. 281–285 (2009).
15. Silva D., Kschischang F.R.: Universal secure network coding via rank-metric codes. IEEE Trans. Inf. Theory **57**(2), 1124–1135 (2011).
16. Tang Z., Lim H.W., Wang H.: Revisiting a secret sharing approach to network codes. In: ProvSec, pp. 300–317 (2012).