情報処理学会研究報告
IPSJ SIG Technical Report

Vol.2009-MBL-50 No.3
2009/9/10

ネットワークコーディングにおける フィールドホッピング

河東晴子[†]　　寺島美昭[†]

ネットワークコーディングの各ノードでの行列演算のフィールドを，送受信者と中継者のみ知る規則に従って切り替えて通信を行うことにより，第三者の傍受に対する耐性を向上するフィールドホッピング方式を提案する.

Field Hopping Scheme in Network Coding

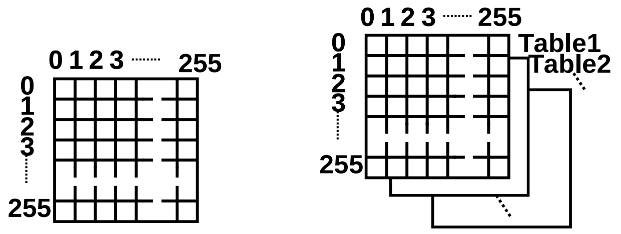
Haruko Kawahigashi[†]　and Yoshiaki Terashima[†]

We propose Field Hopping, techniques for achieving immunity to intercept and interference, by altering the finite field that is a basis of the network coding operation. The field is altered in a predetermined manner known exclusively among the sender, the receiver, and the intermediate nodes.

1. Introduction

The technique of network coding has been successful in many different areas of digital communication in recent years. The two main advantages of the network coding, particularly in mission-critical networks, are saving of the bandwidth through efficient use of the available communication channel and the enhanced data security against stealing of the data at intermediate nodes and channels. In the network coding, each data is transmitted through a channel after linear encoding, so what we see on a channel is an encoded data rather than a raw original data. It is clear that this encoding increases security from a view point of data protection. We propose a new method here to achieve an extra step in this type of data protection.

We first discuss related works on the security aspects of linear network coding and compare ours with them. In [10], the authors consider a sufficient condition which assures linear network coding safe against wiretappers from an information theoretic viewpoint using an extra independent random variable. Instead of the strict information theoretic conditions of [10], more practical consideration using various kinds of random number is made in [11]. The authors of [12] further generalized the work in [10]. A basic idea is to introduce extra measure which prevents an eavesdropper from obtaining any useful information on the transmitted data, based on information theoretic consideration. In [13], the authors consider this problem in combination with another problem of minimizing network cost. A certain measure to deal with the tradeoff of these two matters has been proposed. In [14], an algebraic security criterion, the number of symbols that an intermediate node has to guess in order to decode a symbol, is introduced. In [15], a threat of a malicious node that corrupts the transmission data has been studied, and a new signature scheme for network coding based peer-to-peer file distribution is presented. Our method proposed here deals with different matters of security. Ours does not make wiretapping completely impossible or extremely difficult, but it forces an eavesdropper to make extra efforts and need more time for obtaining useful information. Our analysis uses our previous work [9] on a measure of robustness against wiretapping. Our method is independent from the ones in the above papers, and it is possible to apply our proposal simultaneously together with one of the above papers to enhance security.



(a) Single Filed (Conventional)　　(b) Multiple Field (Proposed)
Figure 1　Finite field operation multiplication tables

In linear network coding, the data transmitted over a network linearly depends on the initial data, and this "linear" operation is performed in a certain finite field, which means a finite set

[†] 三菱電機(株) 情報技術総合研究所
Mitsubishi Electric Corp. Information Technology R&D Center

Vol.2009-MBL-50 No.3
2009/9/10

where one can perform addition, subtraction, multiplication and division with usual rules of operations such as distribution law. By a general theory in mathematics, we know that the size of a finite field is always of the form q^t where q is a prime number and t is a positive integer. It is also known that for any such q^t, we have a certain finite field having the size equal to q^t. It is possible to use any finite field for linear network coding, but it is more convenient to use the size $2^8=256$ or $2^{16}=65536$ because then an element in the finite field is represented with one or two byte data, respectively. We here use the size 256 for simplicity and facility in the actual implementation, but there are more than one field of size 256. Different field structures on a set of size 256 give different operations. For example, the result of 13 times 7 depends on a choice of a field. We propose to use possibility of this choice as an extra enhancement of data security. That is, we propose to change the field structures depending on the source, destination and the time. With this change, even the same inputs give different outputs, depending on the source, destination and the time (see Fig. 1). This clearly brings extra complicacy for the enemy trying to stealing the data through wiretapping.

Field Hopping is a technique for achieving immunity (resilience) to eavesdropping, intercept and interference, by altering the finite field that is a basis of the network coding operation. It can be viewed as a method to spread the information. The field is altered as follows.

(1) The field (the irreducible polynomial) is altered in a predetermined manner known exclusively among the participants, i.e., the sender, the receiver, and the intermediate nodes.

(a) The resilience to eavesdropping increases since the hopping sequence is shared only among the participants.

(b) The resilience to eavesdropping increases as the number of altering polynomials increases.

(2) Alternatively, the polynomial currently in use can be written in the packet

(a)Explicitly write the polynomial in use on the packet e.g. in the header. The resilience to the eavesdropping is small with this method alone, but increases by combination with some other mechanisms.

(b)Alter the polynomial in a predetermined manner shared among the participants, according to the implicitly shared information written in the packet e.g. in the header.

In the rest of this paper, we give a detailed description of this method of switching the field structures and analyze how much extra safety we gain in this method. At first in Section 2, we present a formulation of the linear network coding and the finite field. Then in Section 3, we describe the proposing method and evaluate its effectiveness. In Section 4, we estimate its effectiveness in different environments. In Section 5, we show application examples.

2. Finite Field and Formulation of Linear Network Coding

We present a formal structure of linear network coding. All arithmetic operations are performed within a set of size 256 called a *field*. The operations are given as follows.

First, let GF(2) be the set {0, 1} with operations 0+0=1+1=0, 0+1=1+0=1, 0.0=0.1=1.0=0 and 1.1=1. (Note that the period symbol denotes the multiplication here.) This is a field of size 2. Consider a polynomial of degree m, $f(x)=a_0+a_1x+a_2x^2+\ldots+a_mx^m$, where the coefficients are in GF(2). We can add and multiply two such polynomials in the usual way, but now the coefficients are added and multiplied within GF(2). We say that such a polynomial is *irreducible* if it cannot be decomposed as a product of two such polynomials. We take and fix such an irreducible polynomial $f(x)$ of degree 8 with coefficients in GF(2). (Such examples are explicitly known and not unique.) Then we can define a finite field structure on the set $F=\{0,1,2,3,\ldots,255\}$ as follows. Each number a in F is represented as binary integer expression $a_7a_6a_5\ldots a_0$. We regard this as a polynomial $a_0+a_1x+a_2x^2+\ldots+a_7x^7$ with coefficients in GF(2). If we have two elements a and b in F, we can add the corresponding two polynomials and obtain a new polynomial $c_0+c_1x+c_2x^2+\ldots+c_7x^7$ with coefficients in GF(2). The element c in F given by the binary integer expression $c_7c_6c_5\ldots c_0$ is the sum of a and b. It is easy to see that this c is given by the "exclusive or" of a and b. This is the additive operation in F and the subtraction is defined similarly. Now take two elements a and b in F and consider the corresponding two polynomials with coefficients in GF(2). We multiply these two polynomials in the usual way and obtain a polynomial of degree at most 14. We then divide this polynomial by $f(x)$ and let the remainder be $c_0+c_1x+c_2x^2+\ldots+c_7x^7$. Note that the degree of the remainder is at most 7 since the degree of $f(x)$ is 8. Now the element c in F given by the binary integer expression $c_7c_6c_5\ldots c_0$ is the product of a and b. It is easy to see that we have ordinary rules for addition, subtraction and multiplication such as the distributive law. It can be mathematically shown with the Euclid algorithm that for any a in F, we have unique b in F with $a.b=b.a=1$. This is due to irreducibility of $f(x)$. This b is $1/a$, and a/b is generally defined as the product of a and $1/b$. Then we have all the ordinary rules for addition, subtraction, multiplication and division, thus we have a finite field F of size 256. Note that the value of $a.b$ and a/b depend on the choice of $f(x)$. So if we change $f(x)$ in a way depending on the time and other information, we constantly change the rules of multiplication and division on the same set F of size 256.

Now fix F and we formulate a linear network coding over F, following [2]. Our network consists of a fixed finite directed graph, and our transmission data are sent through this directed graph with certain encoding which occurs at each vertex. For simplicity, we assume

Vol.2009-MBL-50 No.3
2009/9/10

that the graph is acyclic, that is, we do not have a directed loop in the graph. For a small graph, this assumption is rather realistic, and we consider only this situation, but for a large network such as the Internet, we would need a different type of consideration allowing loops in the graph. Then some setting in [2], where they allow loops, is simplified and we have simpler description of the Linear-Code Multicast as follows.

(1) We have a finite directed graph (V, E), where *V* and *E* are the sets of the vertices and edges, respectively. Multiple edges between a pair of vertices are allowed. We have the distinguished vertex *S* called the source vertex.

(2) Transmission data consist of string of elements in the field *F*.

(3) The number of independent paths from *S* to each non-source vertex *T* is called the maximum flow from *S* to *T*, and is denoted by maxflow(*T*). Let *d* be the minimum of maxflow(*T*) for all non-source vertices *T* among *V*.

(4) We send out an arbitrary *d*-dimensional column vector *v* over the base field *F* at the source *S* and it is called the information vector.

(5) Each edge is a channel and transmits an element of the base field *F* as transmitted data.

(6) At the source vertex *S*, we have an $s \times d$ matrix M_S, where *s* is the numbers of the outgoing edges from *S*. We have a matrix multiplication $M_S v$ and send out each entry of this *s*-dimensional column vector through each of the *s* edges starting from *S*. (Each of the *s* rows of the matrix is labeled with the edges from *S*.) The data are sent without any time delay.

$$\begin{pmatrix} y_1 \\ \vdots \\ y_s \end{pmatrix} = M_S\ v = \begin{pmatrix} m_{11} & \cdots & m_{1d} \\ \vdots & \ddots & \vdots \\ m_{s1} & \cdots & m_{sd} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_d \end{pmatrix} \quad (1)$$

(7) At each vertex *T*, we have an $s \times l$ matrix M_T, where *s* and *l* are the numbers of the outgoing and incoming edges at vertex *T*. The rows and columns of the matrix are labeled with the outgoing and incoming edges at vertex *T*. Each incoming edge carries an element of the base field *F* and they give an *l*-dimensional column vector *v*. We have a matrix multiplication $M_T\ v$ and send out each entry of this *s*-dimensional column vector through each of the *s* out going edges without any time delay.

$$\begin{pmatrix} y_1 \\ \vdots \\ y_s \end{pmatrix} = M_T\ v = \begin{pmatrix} m_{11} & \cdots & m_{1l} \\ \vdots & \ddots & \vdots \\ m_{s1} & \cdots & m_{sl} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_l \end{pmatrix} \quad (2)$$

(8) At each of a fixed subset of the vertices having maxflow equal to *d*, we want to recover the original information vector. Such a vertex is called a target vertex.

Note that our data are sent without time delay and our graph has no directed loops, so in Step (7), we can treat each vertex one by one.

We call the $s \times l$ matrix M_T at *T* the encoding matrix at the vertex *T*. For any *d*-dimensional information vector *v* at the source *S*, the data *z* transmitted through an edge *E*, represented by an element in the base field *F*, depends linearly on *v*, so we have a *d*-dimensional row vector w_E such that $z = w_E v$. We call this vector w_E the encoding vector for the edge *E*. For a given graph (*V*, *E*), we call the system of the matrices M_T the encoding scheme.

$$z = w_E\ v = \begin{pmatrix} w_{E1} & \cdots & w_{El} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_l \end{pmatrix} \quad (3)$$

Note that in the above procedure, we have to use the same finite field structure for *F* for data transmission from the source to the destination. However, we can change the finite field structures for *F* as long as a single structure is used for one set of data flows from the source to the destination. This is the main point of our new method we propose here.

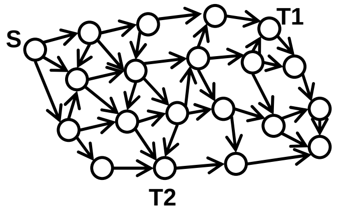


Figure 2 Network Model

3. Switching Irreducible Polynomials and its Effectiveness

It is known that there are 30 irreducible polynomials of degree 8 with coefficients in GF(2). One such example is $x^8+x^4+x^3+x^2+1$. So we have 30 different finite field structures. (If we use a larger unit than 1 byte such as 2 bytes, then we use irreducible polynomials of higher degrees such as 16. Then we have much more choices of irreducible polynomials. For example, the number of irreducible polynomials of degree 9 is 56 and those of degree 10 and 16 are 99 and 4080 respectively.) We propose to change the irreducible polynomials under certain rules.

Before discussing changes of the irreducible polynomials, we deal with a problem of arithmetic operations for a given finite field. One byte data is regarded as a set of coefficients of a polynomial of degree at most 7. Additions and subtractions are both bitwise operations of "exclusive-OR", so it is very easy to perform. Multiplication consists of ordinary multiplication of two polynomials and a division by the irreducible polynomial. The former part is very easy to perform, but the latter part is relatively complicated and time-consuming. So we use a multiplication table beforehand for each irreducible polynomial. One table consists of 256 x 256 bytes of data, that is, 64Kbytes. We need 30 such tables as in Figure 1. For division, it is enough to have a table of $1/x$ for each one byte data x, since y/x is a product of y and $1/x$, so the size of each table is 256 bytes and the number of tables is again 30. (Since 1/0 is not allowed, the size of one table is 255 bytes, strictly speaking.)

We now propose changes of irreducible polynomials. Note that one flow of data from the source to the targets must use the same finite field and that encoding matrices are chosen according to a certain algorithm after fixing a finite field. So there are several options for how to implement such changes and we list some of them as follows. Note that all the nodes must know which finite field they use at present.

(1) Some node in the network chooses a finite field and notifies the other nodes of this choice. Then all the nodes compute the encoding matrices according the same algorithm and share the same system of the encoding matrices. All the nodes continue to use this finite field until the next choice is announced.

(2) Before starting the data transmission, all the nodes agree to have the same rules which determine the finite field in a certain time period. When one time period is over, all the nodes switch to the new finite field under this pre-determined rule.

(3) As a variation of (2), all the nodes agree to have the same rules to determine a sequence of finite fields. After certain times of data transmission, all the nodes switch simultaneously to the next finite field in the pre-determined sequence.

(4) We fix a certain algorithm to generate a sequence of pseudo-random numbers from a given seed number. This algorithm is shared by all the nodes. Then each data packet carries the seed number and the sequential number, such as the 7th number from the seed 135. Then by the pre-determined algorithm, each node can determine which finite field they use.

We now estimate effectiveness of the above proposed method from a viewpoint of data protection.

We deal with data flows at one node. Suppose that a wiretapper can watch incoming and outgoing data and he wants to find out the encoding matrix. This is the type of security threat we consider here. Suppose that the number of the incoming edges is l and that of the outgoing ones is s. Then the incoming data give an l-dimensional column vector x and we multiply the $s\times l$ encoding matrix M_T to get an s-dimensional column vector y of outgoing data. First we assume that the wiretapper knows the finite field and consider how many times of wiretapping are necessary in order to find out the encoding matrix M_T. The wiretapper obtains l times of l-dimensional vectors $x_1,x_2,\ldots,x_l$. Multiplication by M_T to these vectors gives l times s-dimensional column vectors $y_1,y_2,\ldots,y_l$. By putting l times of l-dimensional vectors $x_1,x_2,\ldots,x_l$, we obtain an $l\times l$ matrix X. We similarly obtain a $s\times l$ matrix Y by putting l times s-dimensional column vectors $y_1,y_2,\ldots,y_l$. Then we have $M_TX=Y$, where the left hand side is a product of $s\times l$ matrix and $l\times l$ matrix, and the right hand side is a $s\times l$ matrix. If the determinant of X is not zero (in the finite field), one has the inverse matrix X^{-1}, so can easily find M_T as YX^{-1}. If X does not have an inverse matrix, and one obviously cannot find M_T. In this case, the wiretapper has to continue to steal more data. So we would like to count how many times one has to wiretaps vectors. We have already considered this mathematical problem in a different context in [9], and the answer is

$$f1=\sum_{n=1}^{l}\frac{(2^t)^n}{(2^t)^n-1}=\sum_{n=1}^{l}\frac{256^n}{256^n-1} \quad (4)$$

since the size of the finite field is 256. The computation for this in [9] is basically as follows. The first vector is useful unless it is zero. This probability is (256^l-1)/ 256^l and the expectation of the number of vectors for obtaining the first useful information is 256^l /(256^l-1). The next vector is useful with probability (256^{l-1}-1)/256^{l-1}, so the expectation of the number of vectors for obtaining the next useful information is 256^{l-1}/(256^{l-1}-1). Repeating this process, we obtain the sum as above. Note that the above sum is almost equal to *l*, since 1/256 is small. We called this quantity the *wiretap robustness* (WTR) in [9]. We show the graphs of this quantity *f*1 for different values of *l* and *t* in Figure 3, where *t* is the bit size.

Next we assume that the wiretapper does not know the finite field structure and needs to find it out. Then one proceeds as follows.

(1) One chooses one of the 30 irreducible polynomials and assumes it gives the finite field structure.

(2) One continues as above until determining M_T. Then find another input vector *x* and compare it with $M_T y$ for the output vector *y*.

(3) If the above comparison fails, the irreducible polynomial is not the right one. Repeat this process until 29 out of 30 polynomials are excluded. Then the remaining one gives the right answer and we know the encoding matrix.

We now estimate the number of extra times of wiretapping in the above procedure compared with the case we know the finite field from the beginning. If we wiretap an extra vector, it gives one byte data after multiplications and additions. It gives a correct answer even if we have a wrong finite field structure just by coincidence. Let *p* be 1/256, which is the estimated probability of this coincidence, since one byte gives 256 possible answers. Then $(1-p)^{29}$ is the probability that all the other 29 wrong finite field structures are excluded by such one vector. So with probability $1-(1-p)^{29}$, we need one more vector. Similarly, with probability $1-(1-p^2)^{29}$, we need a further extra vector. The total number of vectors we need in this process is given as follows.

$$f2=1+(1-(1-p)^{29})+(1-(1-p^2)^{29})+\cdots \quad (5)$$

Since *p*=1/256 is small, the third and later terms are negligible, and we obtain 1.11 approximately. In Figure 4, we have a graph showing this quantity *f*2 for various values of *t* for the size 2^t of the finite field. Note that *t* is the number of bits for one element in the finite field and $p=2^{-t}$ in this graph. In the above setting, *l* is typically around 2, 3, or 4, so in these cases, this extra number 1.11 gives an increase of 56%, 37%, and 28% respectively. We have Figure 5 which shows *f*1+*f*2 for different values of *l* and *t*, and Figure 6 showing the increase *f*2 for various values of *l* and *t*, where *t* is as above for the size 2^t of the finite field. Note that this extra increase of efforts for a wiretapper occurs at one node. In network coding, each node transfers coded data, so having the encoding matrix at one node is usually insufficient for obtaining the entire date flow over the network. So a wiretapper needs to continue the same type of wiretapping at other nodes, and this extra increase of the efforts applies to other nodes, too.

In order to determine how frequently we change the finite fields, we need to estimate how frequently a wiretapper can steal the transmitted data at a node. The time interval between changes of finite fields should be (substantially) larger than the product of the above estimate of the necessary number of wiretapping to find the encoding matrix and the estimate of an average time interval between two attempts of wiretapping.

4. Estimates of Effectiveness in Different Environments

We now estimate effectiveness of our proposed method in different settings of network coding.

(1) Random network coding

Recently technique of random network coding has been studied by many researchers [7]. In this scheme, an encoding matrix at each node is not fixed and is chosen randomly at each time, and the encoding vector is sent within each packet. Then at the target node, if one has sufficiently many data with linearly independent encoding vectors, then one can recover the original data.

We now consider switching finite field structures within this framework. In this setting, each node obviously must know which finite field is used, but the wiretapper does not know it. In order for a wiretapper to decode a specific data, which is a *d*-dimensional vector having elements in the finite field, one has to collect *d* times of linearly independent *d*-dimensional encoding vector. Finding the estimate of the number of data we need to collect in order to obtain linear independence is the same mathematical problem as in the last section, so the number is given by *f*1 where the variable *l* is now replaced with *d*, and if one does not know the finite field structure, the estimate of the number of extra vectors we need to find out the finite field is again 1.11 which is given by *f*2 with *p*=1/256 as above. So the framework of random network coding is quite different from the above scheme of fixed encoding matrices, but the mathematical structures and estimates are the same, and Figures 3-6 apply.

Vol.2009-MBL-50 No.3
2009/9/10

(2) Multi-layer network coding

Suppose that the number *l*, the number of incoming data, is 3. Then the extra increase we have with our proposed method is 37%, as shown in the previous section. Now we consider this effect in the setting of multi-layer network coding. In a linear network coding system, one node sends encoded data to other nodes. We can apply the network coding scheme also to these data transmissions through intermediate nodes. In this way, we have double layers of network coding systems. We can consider triple or quadruple layers in a similar way.

In the case of double layers, we can apply our proposed method to both layers. (The two layers can use different finite fields.) Then the estimate of the 37% increase applies to both layers, and a wiretapper has to attack two layers separately. This means that the total increase of the extra efforts for a wiretapper is 88% since $1.37^2=1.88$. Figure 7 shows the effect of this multilayer for different values of *l*

(3) Multi-party use of the same network

Two or more parties can use the same network with network coding, using different finite field structures. Then one party knows only their own data and finite field structure, so they are at the same position as a wiretapper as far as the data of the other party are concerned, and the above analysis applies. We deal with this case from a different viewpoint in the next section.

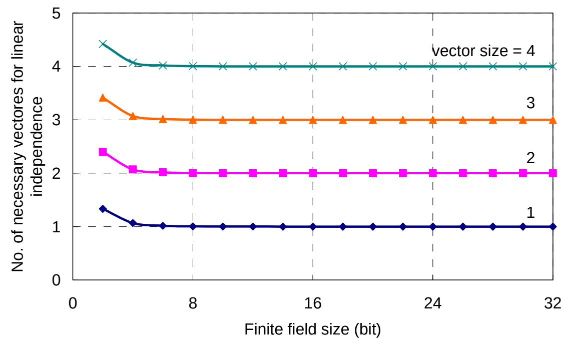


Figure 3 Number of necessary vectors for linear independence: *f*1 (Eq. (4))

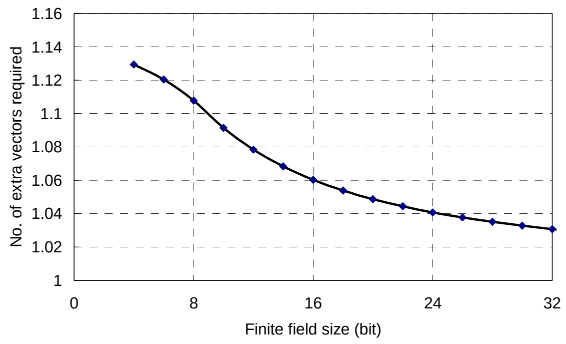


Figure 4 Number of extra vectors required: *f*2 (Eq. (5))

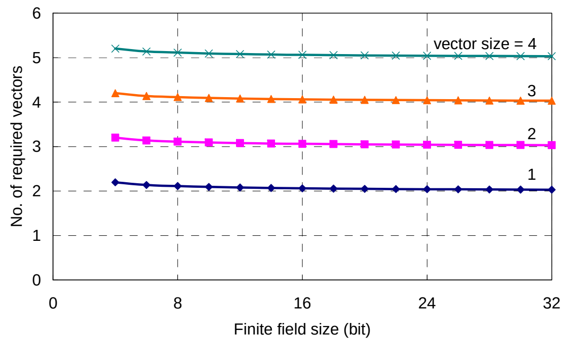


Figure 5 Number of required vectors: *f*1+ *f*2 (Equations (4) + (5))

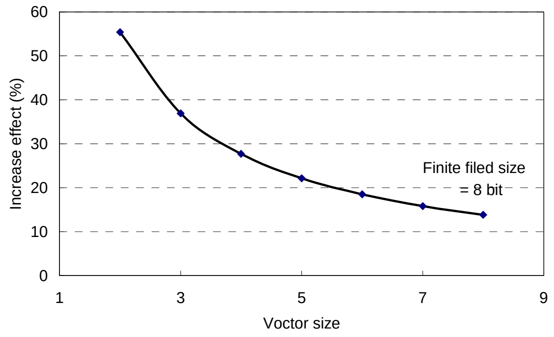


Figure 6 Increase effect in relation to vector size *f*2 / *f*1

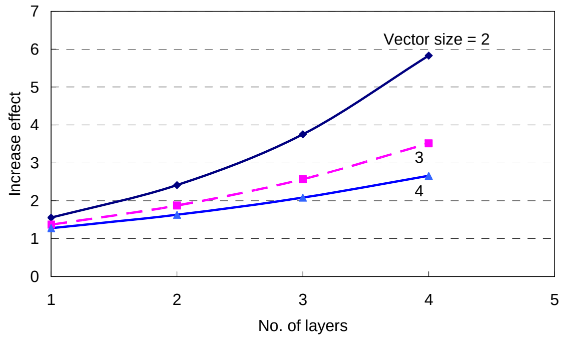


Figure 7 Increase effect in layered coding (Finite filed size: 8 bit)

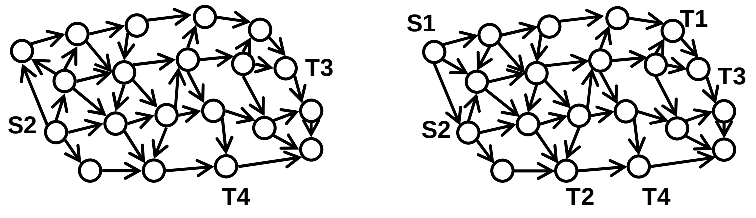
5. Application Examples

We now present application examples of our proposed method. We show a network model in Figure 2 with a source *S* and targets *T*1 and *T*2. Figure 8 (a) shows another network model with a different source *S*2 and targets *T*3 and *T*4. Figure 8 (b) shows a network model where the network models of Figures 2 and 8 (a) are superimposed. These figures represent a case of multi-party communication networks. Figure 2 with source *S* and targets *T*1 and *T*2 represents a network of Party 1, and Figure 8 (a) with source *S*2 and targets *T*3 and *T*4 represents a network of Party 2. When Party 1 and Party 2 transmit data of each party, sharing the same communication equipments at the nodes and the links, they can keep their own data security by using different operation rules, namely, different finite field structures. These finite fields can be either fixed or changed according to the rules known only inside each party. The finite field unique to the party is determined according to the party identification code described in the packet header. Moreover, the multiple parties that pursue their own individual secret communication can switch to start a mutual communication by using the finite field structure. They can still keep their data security against the third party.

This multi-party flexible security mechanism is useful in case of Disaster Relief (DR) where multi organizations are dispatched to the same field. They need to build their own networks in the field with limited number of communication equipments and scarce spectrum of the field. Their primary needs are communication inside their own group and communication to the home office, but they need to share some information with other groups working at the same field, e.g., announcement of damage status of the disaster area. In such kind of mission, security requirement is not as strict as normal operation. They need, however, some kind of security for internal communication of a party.

Using our proposed schemes and network models of Figures 2 and 8, Party 1 corresponds to a dispatched team of one organization and Party 2 corresponds to a dispatched team of another organization.

Our proposed schemes and network models are also effective in case of national Disaster Relief where national rescue teams, civil rescue teams, non-proprietary organizations share Disaster Relief networks.



(a) (b)

Figure 8 (a) A network model with different source and destination, (b) A Superimposed model of two parties

6. Conclusion

In this paper, we have proposed network coding schemes with increased and flexible security by changing finite field structures. Our proposed schemes switch irreducible polynomials used in multiplication of coding operation. This multi-party flexible security mechanism is particularly useful in cases such as Disaster Relief where multi organizations are dispatched to the same field. The multi organizations are able to pursue their own individual secret communication by using different finite field structure, and also able to start a mutual communication by using the same finite field structure while keeping their data security against the third party.

References

[1] R. Ahlswede, N. Cai, S-Y. R. Li, R. W. Yeung, "Network Information Flow," IEEE Trans. on Information Theory, vol. 46, no.4, July 2000, pp. 1204 –1216.
[2] S-Y. R. Li, R. W. Yeung, N. Cai, "Linear Network Coding," IEEE Trans. on Information Theory, vol. 49, no.2, Feb. 2003, pp. 371 –381.
[3] R. Koetter, M. Medard, "An Algebraic Approach to Network Coding," IEEE/ ACM Trans. on Networking, vol. 11, no. 5, Oct. 2003, pp. 782 –795.
[4] T. Ho, M. Medard, R. Koetter, "An Information-Theoretic View of Network Management," IEEE Trans. on Information Theory, vol. 51, no. 4, April 2005, pp. 1295 –1312.
[5] P. Chou, Y. Wu, and K. Jain, "Practical Network Coding," Allerton Conference on Communication, Control, and Computing, Monticello, IL, October 2003.
[6] T. Ho, D. S. Lun, "Network Coding", Cambridge University Press, 2008.
[7] T. Ho, M. Medard, R. Koetter, D. R. Karger, M. Effros, B. Leong, "A random Linear Network Coding Approach to Multicast," IEEE Trans. on Information Theory, vol.52, no. 10, pp. 4413—4430, Oct. 2006
[8] J.-S. Park, M. Gerla, D. S. Lun, Y. Yi, M. Medard, "Code-Cast: A Network-Coding-Based Ad Hoc Multicast Protocol," IEEE Wireless Communication. vol. 13, no. 5, Oct. 2006, pp.76—81.
[9] H. Kawahigashi, Y. Terashima, "Security Aspects of the Linear Network Coding," MILCOM 2007, NCS1-6, Oct. 2007.
[10] N. Cai and R.W. Yeung, "Secure network coding," in IEEE International Symposium on Information Theory, July 2002.
[11] K. Jain, "Security based on network topology against the wiretapping attack," IEEE Wireless Communications, pp.68--71, Feb. 2004.
[12] K. Bhattad and K. Narayanan, "Weakly secure network coding," in Proc. of the First Workshop on Network Coding, Theory, and Applications (NetCod), 2005.
[13] J. Tan and M. Medard, "Secure network coding with a cost criterion," in Proc. of the Second Workshop on Network Coding, Theory, and Applications (NetCod), 2006.
[14] L. Lima, M. Medard, and J. Barros, "Random network coding: A free cypher?," in IEEE International Symposium on Information Theory (ISIT), 2007.
[15] K. Han, T. Ho, R. Koetter, M. Medard and F. Zhaom "On network coding for security," MILCOM 2007, Oct. 2007.