An Integrated Security Mechanism for Network Coding Combining Confidentiality and Integrity

Park, Ji-yong[†], Mi-sun Ryu[†], Jung, Eui Suk[‡], and Seol Dong-Min[‡], and Hong-Shik Park[†]
[†]Information and Communications University, 119 Munjiro, Yuseong-Gu, Daejeon, Korea(ROK).
leosunny, rms0, hspark @icu.ac.kr
[‡]Electronics and Telecommunications Research Institute, 138, Gajungro, Yuseong-Gu, Daejeon, Korea(ROK).
sdm64385, esjung @etri.re.kr

Abstract — **Network coding, whose intermediate nodes decode and encode packets to forward incoming packets, increases the capacity the multicast network. However there are security issues of confidentiality and integrity which degrade and threat network coding. In this paper, we propose Secure Key Checksum (SKC), a secure mechanism using an encryption key, to provide both confidentiality and integrity. To provide the confidentiality, SKC utilizes a block encryption. For the integrity check, SKC uses a checksum based scheme verifying encoded blocks on-the-fly with low computational cost. Also, SKC uses a combined security key for encryption and checksum.**

Keywords — **Network Coding Security, Confidentiality, Integrity, Secure Key Checksum**

I. Introduction

Network coding, first introduced in [1] and developed in [2] and [3], is an in-network data processing technique in order to increase the capacity or the throughput of the network. In network coding, intermediate nodes decodes and encode packets to forward incoming packets. Further development of network coding, called Random Linear Network Coding (RLNC), provides linear operation of encoding, so that it can provide the robustness in packetized networks with lossy links [3]. This new paradigm can be applied to the multicast network, wireless network, ad-hoc network, P2P contents distribution, etc.

Despite the benefits of network coding, there are security issues of confidentiality and integrity [5]. As the intermediate nodes encode and reproduce the outgoing packets, any malicious nodes can pollute the whole network using the network coding, which results in integrity problem. Also, according to the RLNC, confidentiality issue arises as every node in network coding can access incoming information after receiving sufficient packets. These two security issues degrade and threat network coding. The problem caused by integrity is serious as one bad block can corrupt all subsequent blocks in decoding.

This work was partly supported by the IT R&D program of MKE/IITA, [2007-S014-02, Metro-access integrated optical network technology] and MIC, Korea under the ITRC program (C1090-0801-0036) supervised by IITA.

In this paper, we propose an integrated security mechanism with encryption and checksum, which is suitable for network coding. Our mechanism uses a block cipher for encryption and linear operations for checksum. In addition, we use an encryption key as a checksum seed, so that efficient key management is possible.

In section II, we will briefly review the network coding and the problems of network coding security. The security mechanism for network coding is proposed in section III. In addition, simulation results are given in section IV. Finally, we conclude in section V.

II. Network Coding and Security Issues

Network coding is a technology to combine several data packets into one or several output packets. We will discuss about the network coding security issues. Before we go over the network coding mechanism, we will review the current security services of security issues of network coding.

In network coding, each packet is encoded and intermediate nodes do not know the context of transmitted data. However, after collecting the several packets going though, any intermediate node can decode the blocks. Here, we have a confidentiality problem. The confidentiality is a protection of transmitted data from unauthorized disclosure. One of the security mechanisms for confidentiality is the encryption, a mathematical algorithm to transform data into a form which is not readily intelligible.

As mentioned earlier, intermediate node of network coding encodes incoming packets and reproduce outgoing packets. The packets received at the sink node are changed from those sent from a source, and the source cannot guarantee the integrity of sending data. The integrity is an assurance that data received are exactly as sent by an authorized entity, including no modification, insertion, deletion, or replay. The security mechanisms for integrity are digital signature and data integrity. Digital signature is data appended to, or a cryptographic transformation of, a data unit that allows a recipient of the data unit to prove the source and integrity of the data unit and protect against forgery. Also, data integrity is a variety of mechanisms used to assure the integrity of a data unit or stream of data units.

For network coding, there are several researches about security mechanisms. Gkantsidis *et al* [4] proposed a security mechanism using a homomorphic hash function for integrity, but the computation of the homomorphic hash function is heavy. Han *et al* proposed a security mechanism using a digital signature for integrity. However, there is a volume overhead for additional bits for signature. Most of researches are focusing on the integrity problem. However, the confidentiality is a certain problem for network coding

III. Secure Key Checksum

We propose SKC (Secure Key Checksum), an integrated security mechanism suitable for network coding with encryption and checksum, combining an encryption key and a checksum seed. SKC provides confidentiality by using encryption of information with block cipher between source and destination, and integrity by using checksum of linear operation at sink node.

To deal with confidentiality, one of the two problems, we will use an end-to-end encryption. To encrypt information from source node, our mechanism utilizes modification of Practical Network Coding protocol, based on RLNC. In network coding, a packet consists of global encoding vectors and messages. We assume that the source sends a file consisting of h blocks. Each block is encrypted with a block cipher for confidentiality. That is, if the file is $\mathbf{F} = [\mathbf{b}_1, \cdots, \mathbf{b}_h]$, then the encrypted blocks $\mathbf{F'} = [\mathbf{x}_1, \cdots, \mathbf{x}_h]$ are sent to the sink nodes. To send a block of the file through network coding, the block requires the encoding coefficients which carry encoding information of the multiplicative coefficients. Therefore, these encoding coefficients should not be encrypted, and the message is encrypted with keys.

The encrypted output data is the same size as the input data. This make possible that there is no need to change the protocol at the intermediate nodes. One of the examples of block cipher is the Advanced Encryption Standard.

Apart from the typical requirements for encryption mechanisms, such as efficiency and strong security, the only requirement for the encryption of SKC is using the block cipher, whose encrypted output size is the same as the original input. This makes the end-to-end system to choose the state-of-the-art cryptography system with proper requirements.

For another problem, the integrity, checksum check is used. As mentioned above, the encoding and decoding for network coding are linear operations. This forwarding operation of network coding changes the packet sent from the source, and thus existing integrity check methods will not work. To operate with this network coding, the checksum should be based on the linear operation. As a linear operation, our scheme can check the integrity very efficiently. This is suitable for deciding the integrity of incoming packets on-the-fly.

One of the novelties of this thesis is using a key not only for encryption, but also checksum. This reduces the risk of sharing two secrecies.

A. *Procedure of SKC*

Now, we will look in the procedure of SKC. SKC procedure is consist of four steps, the file preparing procedure, the network delivery procedure, the integrity check procedure, and decoding procedure.

The first procedure is a preparation for a file which is to be transmitted. To send a file with encryption, the source and the sink node share an encryption key through a secure channel. Then the source divides a file into blocks and encrypts the blocks. After encryption, the source generates a checksum for each block and sends the checksum to the sink node through the secure channel. Finally, the encoding coefficients are added to the encrypted blocks, which will be sent through network coding.

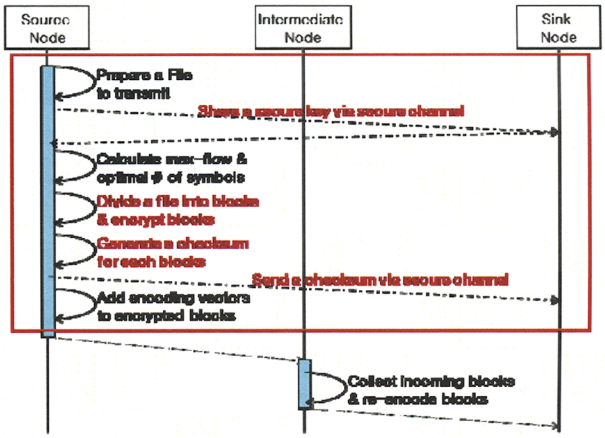

Figure 1. File Preparing Procedure.

The second procedure is the delivery in the network. SKC has no change in the intermediate nodes, so the transmission operation through intermediate nodes is the same as that of usual network coding – encoding and forwarding.

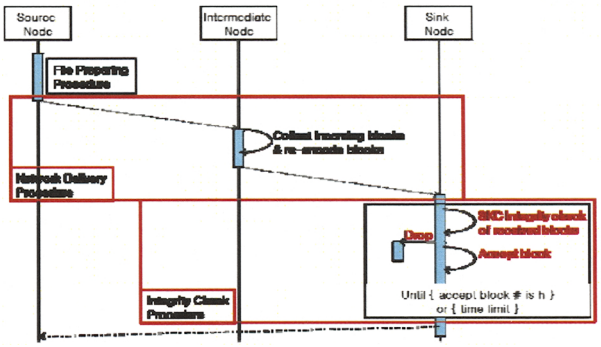

Figure 2. Network Delivery Procedure and Integrity Check Procedure.

The next procedure is integrity check at the sink node. When the packets arrive at the sink node, the receiver, the sink node checks the integrity of incoming packets first. After checking the integrity with linear operations of the key, the checksum, and received packet, the sink decides whether to accept the packet or not – drop the poisoned packets and

accept the validated packets. Then the sink node waits for proper amount of blocks – the same number of max-flow – to decode a file or one generation of a file. If the attacks are too many and not enough packets are gathered for decoding, the sink node requests retransmission of the file (or the generation of a file). After gathering enough blocks, the sink node starts the final procedure, decoding procedure, and decodes to recover the file that is sent by the source.

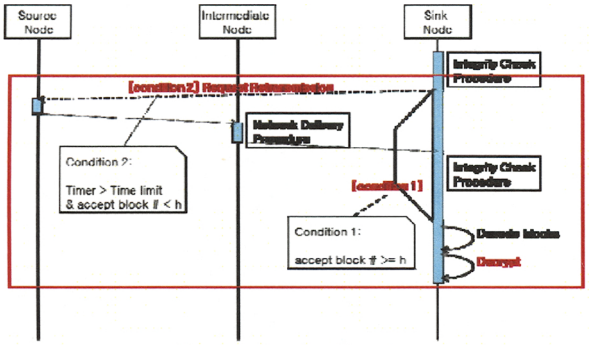

Figure 3. Decoding Procedure.

After receiving enough blocks for decoding, the sink node starts to decode collected blocks. However, when not enough blocks are received at the sink node due to the integrity attack, the sink node requests the source to retransmit the blocks. The source retransmits blocks through network delivery procedure and the sink checks the integrity of received blocks (integrity check procedure). Finally, after receiving *h-a* blocks, the sink node can decode the blocks. When the decoding is successful, the sink node decrypts the decoded blocks to get the original file as the recovered file is an encrypted message.

B. Encryption and Checksum of SKC

Now we will see the procedure of integrity check in detail. To send a packet using SKC encryption, the file is divided into blocks and encrypt the blocks are sent. Then the checksum is calculated for the encrypted blocks. The checksum for a block is the sum of bitwise multiplication of the key and a message block.

Let an encryption key vector $\mathbf{k} = [k_1 \cdots k_N]$, where N is the number of key bits which is same as the bits in a block, and encrypted block i is $\mathbf{x}_i = [x_{i,1} \cdots x_{i,N}]$. Then the checksum for block i is,

$$\begin{aligned} SKC_i &= \mathbf{k} \cdot \mathbf{x}_i^{\mathrm{T}} \\ &= [k_1 \cdots k_N][x_{i,1} \cdots x_{i,N}]^T \\ &= k_1 x_{i,1} + k_2 x_{i,2} + \cdots + k_N x_{i,N} \end{aligned}$$

Also, the checksum of the entire file is a vector of each block checksum $\mathbf{SKC} = [SKC_1 \cdots SKC_h]$.

The sink node in network coding can check the integrity with a shared key and checksum. The packet received at sink node is combination of the global encoding vector and information, i.e.,

$$[\mathbf{g}(e_i) \quad \mathbf{y}(e_i)] = [g_1(e_i) \cdots g_h(e_i) \quad y_1(e_i) \cdots y_N(e_i)]$$

where,

$$\begin{aligned} \mathbf{y}(e_i) &= \mathbf{g}(e_i) \cdot \mathbf{x} \\ &= g_1(e_i)\mathbf{x}_1 + g_2(e_i)\mathbf{x}_2 + \cdots + g_h(e_i)\mathbf{x}_h \\ &= [g_1(e_i)x_{1,1} + g_2(e_i)x_{2,1} + \cdots + g_h(e_i)x_{h,1}, \\ &\quad \cdots, g_1(e_i)x_{1,N} + g_2(e_i)x_{2,N} + \cdots + g_h(e_i)x_{h,N}] \\ &= [\textstyle\sum_{l=1}^{h} g_l(e_i)x_{l,1}, \cdots, \sum_{l=1}^{h} g_l(e_i)x_{l,N}] \end{aligned}$$

At sink node, the key and checksum is shared with the source node. Finally, the sink node has the key, the checksum, the global encoding vector and received information. Then we can check the integrity as follows:

$$\begin{aligned} \mathbf{k} \cdot \mathbf{y}(e_i)^{\mathrm{T}} &= \mathbf{k} \cdot (\mathbf{g}(e_i) \cdot \mathbf{x})^{\mathrm{T}} \\ &= \mathbf{k} \cdot \mathbf{x}^{\mathrm{T}} \cdot \mathbf{g}(e_i)^{\mathrm{T}} \\ &= \mathbf{SKC}_i \cdot \mathbf{g}(e_i)^{\mathrm{T}} \end{aligned}$$

where

$$\begin{aligned} \mathbf{k} \cdot \mathbf{y}(e_i)^{\mathrm{T}} &= [k_1 \ldots k_N][y_1(e_i) \cdots y_N(e_i)]^{\mathrm{T}} \\ &= [k_1 \ldots k_N][\textstyle\sum_{j=1}^{h} g_j(e_i)x_{j,1}, \cdots, \sum_{j=1}^{h} g_j(e_i)x_{j,N}]^{\mathrm{T}} \\ &= \textstyle\sum_{m=1}^{N} k_m \sum_{l=1}^{h} g_l(e_i)x_{l,m} \\ &= k_1(g_1(e_i)x_{1,1} + \ldots + g_h(e_i)x_{1,N}) + \cdots \\ &\quad + k_N(g_1(e_i)x_{1,N} + \ldots + g_h(e_i)x_{h,N}) \\ &= (k_1 x_{1,1} + k_2 x_2 + \ldots + k_N x_{1,N})g_1(e_i) \\ &\quad + \cdots + (k_1 x_{h,1} + \ldots + k_N x_{h,N})g_h(e_i) \end{aligned}$$

and

$$\begin{aligned} \mathbf{SKC} \cdot \mathbf{g}(e_i)^{\mathrm{T}} &= [SKC_1 \ldots SKC_h][\mathbf{g}_1(e_i) \ldots \mathbf{g}_h(e_i)]^{\mathrm{T}} \\ &= (k_1 x_{1,1} + k_2 x_2 + \cdots + k_N x_{1,N})g_1(e_i) \\ &\quad + \cdots + (k_1 x_{h,1} + \cdots + k_N x_{h,N})g_h(e_i) \end{aligned}$$

Let M blocks are receivable from the source and among them, *a* blocks are attacked. After integrity check, the sink node drops attacked blocks and waits until h decodable blocks are received. When *h+a* errorless blocks are received, the sink node can decode the block. However, when attacked packets are so many and the block is not decodable, i.e., *a* > *M-h*, the sink node request retransmission of the block, and receive *h-(M-a)* blocks only.

IV. Simulation

In section IV, NS2 simulation of network coding using SKC is given. The basic network topology is using the NSF network, and call inter-arrival time and call duration time are exponentially distributed. In this network, we alter the percentage of attacked packets from 0.05% to 0.4%.

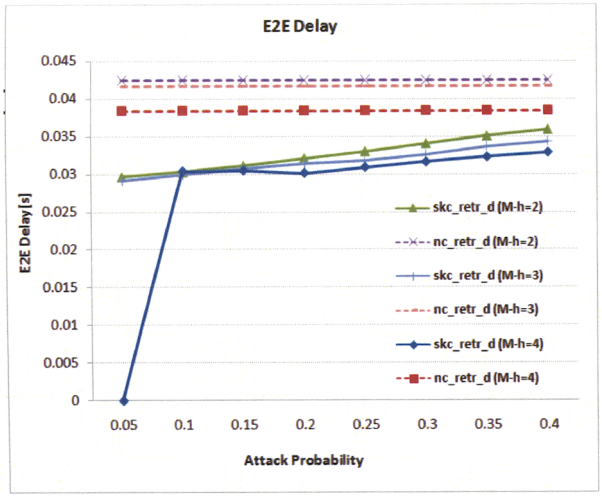

Figure 4. End-to-end delay vs. attack probability.

In figure 4, Dotted lines are the case of basic network coding without any integrity check mechanism, and solid lines are the case of network coding using proposed mechanism, SKC. Also, we change the max-flow value, which represents the attack durability of SKC. Without integrity check, network coding just decodes blocks after gathering h blocks, and requests retransmission when the blocks are not decodable. However, with SKC, attacked blocks are dropped and using other blocks to decode. Therefore we have better end-to-end delay performance.

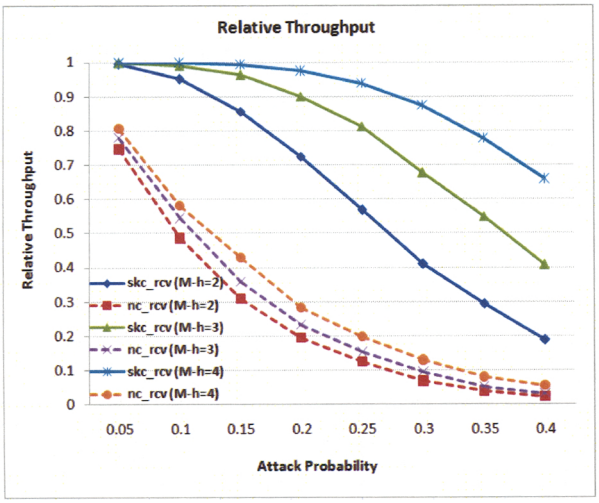

Figure 5. Relative throughput vs. attack probability.

Figure 5 shows the relative throughput of transmission. The relative throughput at the sink node is the ratio of decodable blocks over the number of blocks that source sent. We can see in the graph that network coding using SKC has some attack durability, whereas the network coding without integrity check has serious throughput problem with attack. The decreased throughput requires retransmission of the corresponding blocks.

V. Conclusion

We proposed a security mechanism for network coding, SKC, with encryption and checksum. SKC is an integrated security mechanism for confidentiality and integrity, which is interoperable with other network coding and fast enough to check the integrity on-the-fly. Also, the encryption mechanism is flexible, so that any user requirements can be meet. In addition, we present simulation results using NS2, that the integrity check mechanism is mandatory under the assumption of attacks and the integrity check of SKC is proper for network coding.

References

[1] R. Ahlswede, N. Cai, S. Li, and R. Yeung, "Network information flow," IEEE Transactions on Information Theory, vol. 46, no. 4, pp. 1204–1216, 2000.

[2] P. Chou, Y. Wu, and K. Jain, "Practical network coding," Allerton Conference on Communication, Control, and Computing, Allerton, USA, September 2003.

[3] T. Ho, M. M´edard, R. Koetter, D. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast,"IEEE Transactions on Information Theory, vol. 52, no. 10, pp. 4413–4430, 2006.

[4] C. Gkantsidis and P. Rodriguez, "Cooperative security for network coding file distribution," In Proc. of IEEE INFOCOM, Barcelona, April 2006.

[5] N. Cai and R. Yeung, "Secure network coding," IEEE International Symposium on Information Theory, Lausanne, Switzerland, July 2002.

[6] S.-Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," IEEE Transactions on Information Theory, vol. 49, no. 2, pp. 371–381, 2003.

[7] Han, Keesook, Ho, Tracey, Koetter, Ralf, Medard, Muriel, and Zhao, Fang, "On network coding for security," Military Communications Conference, 2007. MILCOM 2007. IEEE, October 2007