2006－CSEC－34 (49)
2006／7／21

# ネットワークトポロジを考慮した効率的なマルチキャスト暗号方式

福島 和英[†] 清本 晋作[†] 田中 俊昭[†] 櫻井 幸一[††]

[†] 株式会社KDDI研究所 〒356-8502 埼玉県ふじみ野市大原2-1-15
[††] 九州大学大学院システム情報科学研究院 〒812-8581 福岡県福岡市東区箱崎6-10-1
E-mail: †{ka-fukushima,kiyomoto,tl-tanaka}@kddilabs.jp, ††sakurai@csce.kyushu-u.ac.jp

あらまし 有料コンテンツのマルチキャスト配信サービスは，近い将来，個人PCや携帯端末向けの主流サービスとして期待されているが，コンテンツの著作権保護方法の確立が大きな課題となっている．海賊版コンテンツの流通を防ぐための手段の1つとして，コンテンツを暗号化しておく方法があり，これらの暗号化鍵を管理するための効率的な鍵管理方式が求められている．これまでにも，マルチキャストシステム用の鍵管理方式は考案されてきたが，いずれの方式も単純な鍵管理構造を用いており，鍵管理効率の最適化については十分な検討がなされていなかった[1]～[3]．そこで，我々は，クライアントの能力に応じて柔軟な鍵管理が実現できる方式である，$\tau$-鍵管理方式を提案した[4]．さらなる鍵管理方式の最適化を行うためには，ネットワークトポロジを考慮した鍵管理構造を構成することが不可欠となる．本稿では，マルチキャストに適合する鍵管理方式を提案する．さらに，本方式において鍵管理ルータに必要とされる計算量および領域量を定量的に評価し，これらの最適化について議論する．

キーワード マルチキャスト暗号、鍵管理、グループ通信、コンテンツ配信

# Key-Management Scheme for Secure Multicast based on Network Topology

Kazuhide FUKUSHIMA[†], Shinsaku KIYOMOTO[†], Toshiaki TANAKA[†], and Kouichi SAKURAI[††]

[†] KDDI R&D Laboratories Inc., 2–1–15, Fujimino-shi, Saitama, 356–8502 Japan
[††] Faculty of Information Science and Electrical Engineering, Kyushu University, 6–10–1, Hakozaki, Fukuoka-shi, Fukuoka, 812–8581 Japan
E-mail: †{ka-fukushima,kiyomoto,tl-tanaka}@kddilabs.jp, ††sakurai@csce.kyushu-u.ac.jp

**Abstract** Pay-multicasting services are expected to be a main service for mobile devices and personal computers in near future, and copyright protection is a major issue for the services. The encryption of digital content is one solution to prevent illegal copying and generate income from clients. Thus, we proposed $\tau$-key-management scheme which provides flexible lock management according to the capacity of a client was proposed. However, this scheme uses a logical key-management structure, and does not consider a network topology. Some key-management systems for multicast systems have been proposed [1]～[3]. However, these schemes use simple key-management structure, and sufficient examinations for optimization of them have not been carried out. Then, we proposed the $\tau$-key-management scheme which provides system which provides flexible key-management based on computational capacities of clients [4]. Additionally, a key-management scheme based on a network topology is required for further optimization. This paper proposed a key-management suited to a multicast system Then, we show the quantitative computational cost and storage cost on key-management routers using five elements; that is the total number of clients, the average service usage time, the duration of keys update, the degree of key-management tree, and the maximum number of clients in a group. Finally, we discuss optimization of our scheme.

**Key words** Multicast Encryption, Key Management, Group Communication, Content Distribution

# 1. Introduction

## 1.1 Background

Recently, high-speed Internet has been expanded to include 3G mobile services and FTTH service for personal computers. Digital content delivery services that provide music, movies, and games for mobile phones have been major services, and pay-multicasting services are expected to be a new service of particular importance in the near future. However, copyright protection is a major issue for these services. The duplication of digital content is easy and requires little effort. As a result, illegal content circulates widely. The encryption of digital content is one solution to prevent illegal copying and generate income from clients. Many schemes for managing encryption keys have been proposed. These schemes are called *key-management schemes.* In large-scale services, joins and leaves of clients occur frequently and the total number of managed keys is enormous. Computational cost of clients is too high not only for mobile devices with low capacities but for personal computers in these services. Thus, secure and efficient key-management schemes are required.

## 1.2 Related Works

Existing key-management schemes are classified into two types: *stateless schemes* and *stateful schemes.* In a stateless schemes [5]~[13], the content encryption key is based only on the content distribution message and the pre-shared key of each client; keys for each client are never updated. In a stateful scheme [1]~[3], [14]~[16], content encryption key is based on the content distribution message and the old content encryption key and old key encryption keys; these keys can be updated. We do not need to consider the maximum number of clients in this scheme. Additionally, the size of a message does not depend on the number of clients who leave the service. Thus, stateful schemes are suitable for long-term or large-scale services. The simplest stateful scheme is a star-based scheme [1]. All clients use the shared key, and the key is updated when a client joins or leaves. Wong *et al.* [2] and Wallner *et al.* [3] proposed a tree-based scheme. The shared key and the key encryption keys are assigned to the root node and the interior nodes respectively. Additionally, individual keys of clients are assigned to leaf nodes. Each client has keys assigned to all nodes along the shortest path from the root node to the leaf node.

Star-based scheme and tree-based scheme are detailed in below. Each client shares an individual key with the key-management server and trusts the server. And the shared key is updated in join process and leave process.

### 1.2.1 Star-based Scheme

A client has two keys, a the shared key and an the individual key.

**(1) Join Process**

The server updates the current shared key. Then it encrypts the updated shared key with the old shared key and sends it to existing clients. Additionally, the server encrypts the key with the individual key of a new client, and sends it to this client.

**(2) Leave Process**

The server updates the current shared key. Then it encrypts the updated shared key with each individual key and sends it to each client.

### 1.2.2 Tree-based Scheme

This scheme uses the $k$-ary tree. The shared key is assigned to the root node of the tree, individual keys of the client are assigned to leaf nodes, and key encryption keys are assigned to intermediate nodes. A client has $\lceil \log_k N \rceil$ keys assigned to the ancestor nodes of the node where its individual key is assigned.

**(1) Join Process**

The server updates the shared key and all the key encryption keys assigned to the ancestor nodes of the node where the individual key of a new client is assigned. Then, it encrypts each updated keys with the old keys and sends them to existing ent clients who have the old keys. Finally, it encrypts these keys with the individual key of the new client, and sends the encrypted keys to this client.

**(2) Leave Process**

The server updates the shared key and all the key encryption keys assigned to the ancestor nodes. Then, it encrypts all the updated keys with keys assigned to the child nodes, and sends them to existing clients who have the child keys. Finally, it encrypts these keys with the individual key of the new client, and sends the encrypted keys to this client.

## 1.3 Our Contribution

This paper propose a key-management suited to a multicast system Then, we show the quantitative computational cost and storage cost on key-management routers using five elements; that is the total number of clients, the average service usage time, the duration of keys update, the degree of key-management tree, and the maximum number of clients in a group. Finally, we discuss optimization of our scheme.

# 2. Multicast System

Multicast is a bandwidth-conserving technology which reduces traffic by simultaneously delivering single data to multiple clients. Applications which take advantage of multicast include distribution of digital content such as music, movie, software, and so on. Multicast system requires special routers for multicast forwarding. We proposed a key-management scheme for multicast system under the following assumption.

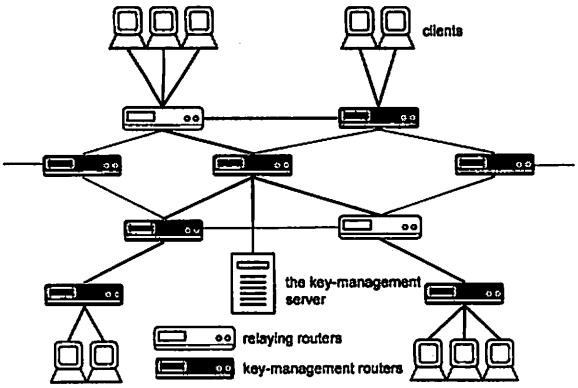

Figure 1 Our multicast architecture with key-management routers

- All routers in the system supports multicast forwarding
- Distribution tree for the key-management server does not change
- One or more key-management routers exist in the path from the key-management server to a clients

A key-management router is a router which has the following functions:

- A key-management router can encrypt and decrypt a key
- A key-management router can securely store keys
- A key-management router can securely communicate with other routers
- A key-management router can play the role of a relaying routers which is a general routers supporting multicast forwarding.

Figure 1 shows our multicast architecture. Thick lines indicate connections are contained in the distribution tree. Distribution tree can be constructed by multicast routing protocols such as Core-Based Tree (CBT) [17], PIM-DM [18], PIM-SM [19] and so on. Figure 2 shows the distribution tree extracted from Figure 1.

The router nearest to the key-management server is denoted by root router, the key-management routers nearest clients by leaf routers, and the other key-management routers by intermediate routers. The key-management server issues instructions of update of keys to the root router. The root router corresponds to the root node of a key management tree in tree based scheme, and stores the shared key. Intermediate routers correspond to intermediate nodes of the tree and stores key encryption keys. Finally, the leaf routers correspond to leaf node of the tree and stores group keys and individual key of clients belong to the groups which are managed by the leaf routers.

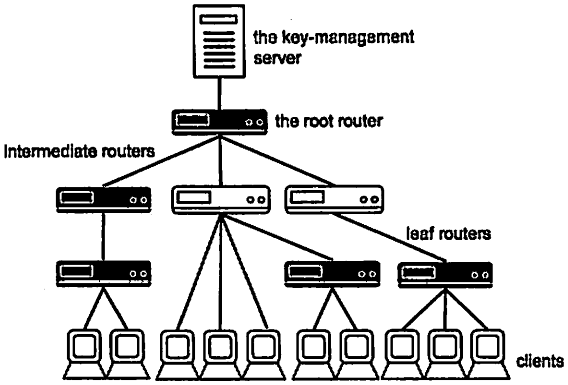

Figure 2 Distribution tree extract from Figure 1

## 3. Proposed Scheme

$\tau$-key-management scheme [4] is a centralized scheme and the key-management server undertakes all the key update process. In our assumptions, key-management routers can store keys and update them itself. Thus, we modify this scheme into a distributed scheme which makes routers share the load. Our scheme consists of the following five processes: preparation, join, leave, content multicast, and keys update.

### (1) Preparation Process

The process is executed only once; when the key-management scheme is started.

**Step 1 Make Groups**

Each leaf router generates a group consist of client which are connected to the router.

**Step 2 Construct Key-management Tree**

The route node generates a shared key and distributes it to all routers. Next, each intermediate router generates a key encryption key and distributes it to descendant nodes. Each leaf router generates a group keys which is shared among all the clients connected to the router. Then, a leaf router encrypts the group key with each individual key of each client and sends it to each client.

### (2) Join Process

Routers do not update keys. The leaf router connected to a new client encrypts the shared key with the individual key of a joining client, and sends it to the client.

### (3) Leave Process

Routers do not update keys. However, the leaf routers from which the client left set a flag. The flag indicates that the group key is going to be updated in the next keys update process.

### (4) Content Multicast Process

A content provider sends content to the key-management server. The content is encrypted with the shared key, and sends it to all the clients.

(5) **Keys Update Process**

Keys are upgraded every $\tau$ ($\tau > 0$) seconds. The routers update the keys according to the following steps:

**Step.1 Update of Keys on Leaf Routers**
Leaf routers on which flags are set update group keys. These leaf routers encrypt the updated group keys with an individual key for each client, and send it to each client. Then, these routers send a key update request to the parent router.

**Step.2 Update of Keys on Intermediate Routers**
Intermediate routers, which receive the request from the child routers, update key encryption keys. Next, these routers send the updated key and a key distribution request to the child routers. Then, the child routers encrypt the updated key with the key encryption key assigned to the router and distribute the key to all the child routers. Finally, these routers send a key update request to the parent router.

**Step.3 Update of Keys on the Root Routers**
The root router receives the request from the child routers and updates key encryption keys. Next, the root router sends the updated shared key and a key distribution request to the child routers. Then, the child routers encrypt the updated shared key with the key encryption key assigned to the router and distribute the key to all the child routers.

## 4. Efficiency of Our Scheme

We evaluate computational cost and storage cost on key-management routers. Computational cost, *Comm*, is defined by the maximum number of distinct encrypted keys which a router issues per second. Then, storage cost, *Stor*, is defined by the maximum number of keys which a client receives per second.

We evaluate our scheme under the following assumptions:

- A client joins or leaves based on service model *VMM* (detailed in below)
- Each leaf router manages $m$ clients
- The shared distribution tree is a complete $k$-ary tree and the number of leaf router is $\frac{N}{m}$

We introduce a service model, *Variety Multicasting Model (VMM)*. This is a model of general multicasting services where a client randomly joins and leaves. Taninaka *et al.* [20] simulated the same service model to evaluate their key-management scheme. *VMM* is a multicasting model which satisfies the following three conditions.

- The total numbers of clients using the service is constant $N$ (in initial and steady state).
- A join of a client is according to Poisson process with intensity $\lambda = \frac{N}{T}$.
- A leave of a client is according to Poisson process with intensity $\lambda = \frac{N}{T}$.

Where $T$ is the average service utility time.

In our scheme, a group key of a group where a client leaves is revoked. Thus, the number of revoked group keys is $\frac{N}{T}$ in *VMM*.

The server issues encrypted keys when (1) join process and (2) keys update process are executed. Each router must hold all the keys assigned to the ancestor nodes. Furthermore, leaf routers must hold individual keys of clients which are connecting to the router. Thus,

$$Stor = \left\lceil \log_k \frac{N}{m} \right\rceil + m.$$

(1) **Join Process**

$\frac{N}{T}$ clients join per second, and each client belongs to one of $n = \frac{N}{m}$ groups. Thus, a leaf router issues $\frac{N}{nT} = \frac{m}{T}$ *distinct* encrypted keys to joining clients per second.

(2) **Keys Update Process**

**Step.1 Leaf Routers**
The leaf routers encrypt the updated group key with $m$ individual keys of clients which are connecting to the group.

**Step.2 Intermediate Routers**
Intermediate routers encrypted the update key of the parent node with a key encryption key assigned to the router.

**Step.3 The Root Routers**
The root router updates the shared key. However, the router let child routers to encrypt and distribute the updated shared key.

Thus, leaf routers where a client leaves, issues most encrypted keys, and

$$Comm = \max\{m, 1, 0\} = m.$$

## 5. Optimization method

### 5.1 Overview of Optimization

We optimize our scheme. Computational cost and storage cost on key-management routers in this scheme depend on the following five elements:

- The total number of clients $N$
- The average service usage time $T$
- The duration of keys update $\tau$
- The degree of the key-management tree $k$
- The maximum number of clients in a group $m$

The key distributor cannot control the parameters $N$, $T$, $\tau$ and $k$. The parameters $N$ and $T$ depend on clients action, $\tau$ depends on policies of a content provider, and $k$ depends on the network topology. The distributor can control only the parameters $m$. Here, we consider the following two elements:

(1) **The duration of keys update: $\tau$**

This parameter should be controlled by the content provider, since this parameter indicates the period that a client can watch content free of charge. For example, the content provider can set $\tau$ to a few seconds or a few tens of

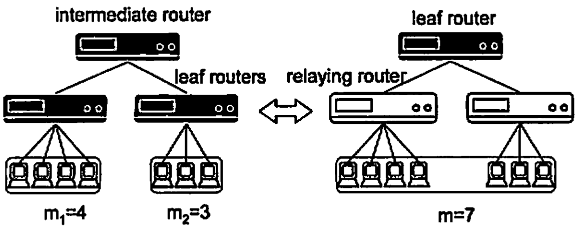

Figure 3 Change of the number of clients in a leaf router

seconds. The content provider will not mind even if clients watch free content for periods exceeding a few tens of seconds in a service that charges for every minute.

(2) **The number of clients in a leaf router: $m$**

It is difficult to control this parameter completely. However, this parameter can be changed to a certain degree by changing the key-management structure. We can increase the number of clients in a leaf router by making the parent router execute group manage instead of multiple leaf routers. As a result, these leaf routers are to play only the role of communication relaying. Contrarily, we can decrease $m$ by making the child routers execute group.

### 5.2 Optimal Parameter Setting for $m$

Both computational cost and storage cost on key-management routers increase as the parameter $m$ increase. Thus, optimal parameter setting for $m$ is the smallest possible value. That is, routers directly connected to clients should play the role of leaf router.

## 6. Conclusion

This paper proposed a key-management suited to a multicast system Then, we showed the quantitative computational cost and storage cost on key-management routers using five elements; that is the total number of clients, the average service usage time, the duration of keys update, the degree of key-management tree, and the maximum number of clients in a group. Finally, we discussed optimization of our scheme.

In our future work, we are going to show the detailed optimization method for proposed scheme.

### References

[1] M. Burmester, "On the risk of opening distributed keys.," Advances in Cryptology - CRYPTO '94, Lecture Notes in Computer Science 839, pp.308–317, 1994.

[2] C.K. Wong, M.G. Gouda, and S.S. Lam, "Secure group communications using key graphs," Proc. of the ACM SIGCOMM '98 conference on Applications, technologies, architectures, and protocols for computer communication, pp.68–79, 1998.

[3] D. Wallner, E. Harder, and R. Agee, "Key management for multicast: Issues and architectures." RFC 2627 (Informational), 1999.

[4] K. Fukushima, S. Kiyomoto, and T. Tanaka, "Evaluation of dual-structure key-management scheme suitable for mobile services," Proc. of The 7th International Conference on Mobile Data Management (MDM'06), 2006.

[5] M. Naor and B. Pinkas, "Efficient trace and revoke schemes.," Financial Cryptography 2000, Lecture Notes in Computer Science 1962, pp.1–20, 2000.

[6] D. Naor, M. Naor, and J.B. Lotspiech, "Revocation and tracing schemes for stateless receivers," Advances in Cryptology - CRYPTO 2001, Lecture Notes in Computer Science 2139, London, UK, pp.41–62, Springer-Verlag, 2001.

[7] D. Halevy and A. Shamir, "The lsd broadcast encryption scheme.," Advances in Cryptology - CRYPTO 2002, Lecture Notes in Computer Science 2442, pp.47–60, 2002.

[8] Y. Dodis and N. Fazio, "Public-key broadcast encryption for statless receivers," Digital Rights Management (DRM '02), Lecture Notes in Computer Science 2696, 61-80.

[9] T. Asano, "A revocation scheme with minimal storage at receivers.," Advances in Cryptology - ASIACRYPT 2002, Lecture Notes in Computer Science 2501, pp.433–450, 2002.

[10] T. Asano, "Reducing storage at receivers in sd and lsd broadcast encryption schemes.," Information Security Applications, 4th International Workshop, WISA 2003, Lecture Note in Computer Science 2908, pp.317–332, 2003.

[11] T. Asano, "Secure and insecure modifications of the subset difference broadcast encryption scheme.," Information Security and Privacy: 9th Australasian Conference, ACISP 2004, Lecture Note in Computer Science 3108, pp.12–23, 2004.

[12] T. Asano, "Reducing receiver's storage in cs, sd and lsd broadcast encryption schemes.," IEICE Transactions, vol.E88-A, no.1, pp.203–210, 2005.

[13] J.H. Cheon, N. Jho, M.H. Kim, and E.S. Yoo, "Skipping, cascade, and combined chain schemes for broadcast encryption." Cryptology ePrint Archive, Report 2005/136, 2005.

[14] I. Chang, R. Engel, D. Kandlur, D. Pendarakis, and D. Saha, "Key management for secure internet multicast using boolean function minimization techniques," Proc. IEEE Infocomm'99, pp.689–698, 1999.

[15] Y. Kim, A. Perrig, and G. Tsudik, "Simple and fault-tolerant key agreement for dynamic collaborative groups," ACM Conference on Computer and Communications Security, pp.235–244, 2000.

[16] B. Pinkas, "Efficient state updates for key management," Workshop on Security and Privacy in Digital Rights Management 2001, Lecture Notes in Computer Science 2320, pp.40–56, 2001.

[17] A. Ballardie, "Core based trees (cbt) multicast routing architecture, rfc2201," 1997.

[18] D. Estrin, D. Farinacci, A. Helmy, D. Thaler, S. Deering, M. Handley, V. Jacobson, C. Liu, and L. Wei, "Protocol independent multicast-sparse mode (pim-sm): Protocol specification, rfc2362," 1998.

[19] A. Adams, J. Nicholas, and W. Siadak, "Protocol independent multicast - dense mode (pim-dm): Protocol specification (revised), rfc3973," 2005.

[20] Y. Taninaka and M. Yamamoto, "A key management protocol with active network technology for secure multicast," Proc. of The 2nd International Workshop on Active Network Technologies and Applications (ANTA 2003), pp.63–74, 2003.