When DRM Meets Restricted Multicast $\cdots$

—- A Content Encryption Key Scheme for Multicast Encryption and DRM

Min FENG
Microsoft Research Asia
Beijing, P.R.China
Email: minfeng@microsoft.com

Bin ZHU
Microsoft Research Asia
Beijing, P.R.China
Email: binzhu@microsoft.com

Abstract— **In many applications it is desired to save the content received from restricted broadcast or multicast to local files for subsequent replays. The locally saved files should be protected by a Digital Rights Management (DRM) system to prevent unauthorized usage. It is a great challenge to combine DRM with restricted broadcast or multicast since they are designed for different applications. In this paper, we first present two straightforward solutions and discuss their drawbacks. We then present a novel content encryption key scheme for restricted broadcast and multicast that facilitates subsequent DRM protection for the saved content. It enables direct saving of encrypted content to local files and easy generation and management of decryption keys for replays of saved files. Only a single key needs to be delivered to a client in a license. Security of the proposed key scheme is analyzed, and comparison of the three methods in also discussed in this paper.**

I. INTRODUCTION

In premium broadcast and multicast applications such as pay-TV or pay-multicast services, content is encrypted. The main focus of these applications is to restrict access to the content to a privileged group of users. Users not belonging to the group are unable to decrypt the protected content. An access control system is used to achieve the goal. In these applications, there exists a center which is responsible for generating and distributing the content encryption keys to the targeted group of users. Such a group is dynamic. Users can enter or leave the group. To ensure forward and backward access control, the content encryption key has to be changed every time when there is a change in membership of the group. The frequency to change the content encryption key depends on a specific application. It may be relatively infrequent such as the next billing period or the next pay-per-view event. It may also be as frequent as every few seconds [1]. Many schemes have been developed to transmit the content encryption keys securely and efficiently to the legitimate users so that only the privileged users can access the content encryption keys. A partial list of the papers is [2]-[11]. Some of these schemes study broadcast encryption. Others study multicast encryption. Generation of content encryption keys is not the focus of these schemes. Any method can be used to generate a new content encryption key, as long as the new key cannot be predicted from the old keys.

Unlike the access control used in restricted multicast and broadcast, Digital Rights Management (DRM) is a system that provides persistent managements of all the rights ranging from description, identification, trading, and protection to monitoring and tracking for digital contents from creation to consumption [12], [13]. There are several commercial DRM systems available on the market. A typical one is the Windows Media Rights Manager (WMRM) from Microsoft [14]. Standardization organizations have also been working on DRM standards. The Open Mobile Alliance (OMA) has adopted a DRM standard recently for mobile environments [15]. MPEG has adopted a DRM standard called the Intellectual Property Management and Protection [16]. The Digital Media Project (DMP) is working on an open DRM standard that ensures interoperability [17]. In a typical DRM system, the content publisher encrypts the digital media and then packages the content into a content object. The decryption key is encrypted and stored in a rights object called a license. A license also contains a specification of the acquired rights. It is usually locked to the targeted device to prevent a user from sharing with other unauthorized devices. Licenses and protected content objects are usually distributed and stored separately to make it easier to manage the entire system. To consume a protected content object, a user has to first acquire a license containing the decryption key along with the acquired rights from the license server. A DRM system enforces acquired rights through the trusted DRM modules on the client side.

In many applications, a user is allowed to record and replay the content from restricted broadcast or multicast, with possible additional payment. Certain restrictions may be associated with the recorded content. For example, the recorded content may be restricted to be playable within a week of recording and is not allowed to play on other devices. DRM is a perfect solution to guarantee the recorded content is consumed according to the specified rights. That means that a DRM system needs to be applied after restricted broadcast or multicast. Combining the access control in restricted broadcast or multicast with DRM is a great challenge, esp. when the content encryption key changes frequently, since the two systems are designed with different goals in mind. The access control in restricted broadcast or multicast is designed to ensure that the users not belonging to a specific group are unable to access the privileged content. It has to ensure that a user who joins (leaves) the specific group cannot consume the content before (after) he or she joins (leaves) for backward (forward) access control. This is archived by changing the content encryption key, as mentioned above. In some applications, content encryption keys are required to change periodically no matter there is a membership change or not for the specific group. These key changes can occur in the middle of playing. In a DRM system, on the other hand, a content object is usually encrypted with a single encryption key. There is no need to rekey in encrypting a content object in DRM. This approach greatly simplifies the key management and generation by a license server and key delivery to users. For example, in Microsoft's WMRM, a license server needs to store and remember only a seed. There is no need to use a database for a license server. During license acquisition, a client sends to the license server the DRM header extracted from the protected content which contains a key ID. The license server generates the content encryption key from the received key ID along with the seed it knows, and packages the key in a license to deliver to the client. A single key is contained in the license. Direct saving of the encrypted content from restricted broadcast or multicast will require multiple, possibly a substantial number of content encryption keys for a saved content object. This would produce a great burden for a license server to manage and generate content encryption keys for content objects. A large database may be required to handle the key generation during users' license acquisition. The size of a license may also be dramatically increased in order to deliver all the necessary decryption keys to a client.

In this paper, the issues associated with applying a DRM system to protect the recorded content from restricted broadcast or multicast are addressed. First we describe two straightforward solutions and discuss their drawbacks. Then we propose a novel key scheme based on RSA [18] to generate content encryption keys for broadcast or multicast encryption. The key scheme simplifies dramatically the key management and delivery in the DRM system subsequently applied to the recorded content. Security of the proposed key scheme as well as its advantages and disadvantages are also discussed. Encrypted media from broadcast or multicast is directly saved in this proposed scheme. Only a single key is contained in a license for a recorded content object. The proposed scheme is secure to collusion attacks in which clients with knowledge of different content encryption keys work together attempting to derive new keys that the group of users do not know. It has an additional advantage that multiple recorded files can be merged together directly into a single file by the client, and only a single key is delivered in a license to decrypt the merged file, no matter the merged file contains consecutive encryption blocks or not.

We would like to point out that the novel key scheme to be presented in this paper is the content encryption key scheme, i.e., a scheme to generate content encryption keys. It has nothing to do with the key schemes used in broadcast or multicast encryption, where a key scheme is designed to ensure that only the users belonging to a specific privileged group can receive the new content encryption key when rekeying occurs. Our content encryption key scheme can be applied together with any broadcast or multicast encryption schemes.

As far as the issues addressed in this paper concern, there is no need to differentiate restricted broadcast from restricted multicast. For simplicity, we would use multicast in the remaining of the paper to refer to both broadcast and multicast. Similarly, multicast encryption would refer to both broadcast encryption and multicast encryption.

The remaining of the paper is organized as follows. In Section II, two straightforward schemes are presented and discussed. The novel content encryption key scheme is presented in Section III. In Section IV, we analyze security of the proposed key scheme. Comparison of the three described schemes are offered in Section V, followed by the conclusion in Section VI.

II. Two Straightforward Schemes

Before describing the two straightforward solutions to the problem addressed in this paper, we would like to define the term *encryption block* or simply *block*, which means a block of data encrypted with the same content encryption key in broadcast or multicast. Auxiliary data assisting decryption of the encrypted data such as the encryption block ID may also be included in the encryption block. In restricted multicast, distributed content consists of a sequence of encryption blocks.

The first straightforward solution, referred to as the *direct recording* method, saves encryption blocks received from multicast directly to local storage. A user needs to acquire a license from the license server to play the saved content. A license in this method has to contain all the content encryption keys associated with all the possible combinations of saved encryption blocks. That means that a license may contain multiple, possibly a substantial number of, content encryption keys. The license server has to store *all* the content encryption keys used in multicast in order to deliver necessary keys required by users.

The second straightforward solution, referred to as the *transcryption* method, works in a similar way as a transcoder to compression: a receiver transcrypts the multicast encryption into DRM encryption. In this method, a receiver first decrypts the encrypted content from multicast encryption where multiple content encryption keys are used, and then re-encrypt the content with the DRM encryption where a single content encryption key is used for each saved file. In addition to the disadvantage of computational overhead during the transcryption, security of the system is also lowered. In this method, the client is required to perform content packaging that a content publisher does in a conventional DRM system. Since users are not trusted in the threat model of DRM applications, more client modules need to be protected against hacking and reverse engineering. In this method, the DRM content encryption key needs to be acquired from the license server by the client before re-encryption. This ensures that the recorded content can be played with other devices – another device only needs to acquire a license from the license server to play the recorded content[1].

[1]We have assumed here that a license is locked to a device.

III. An Integrated Content Encryption Key Scheme

A scheme better than the two schemes described in Section II is to design a key scheme to generate content encryption keys for multicast encryption by taking into account the specific requirements of the DRM applied to the recorded contents. In this section, we present a novel integrated key scheme designed to meet the needs of both multicast encryption and DRM. The scheme is based on RSA [18]. It contains the following four processing phases.

Setup: The center of multicast first generates two large prime numbers p and q. Let $n = p \cdot q$ and $\phi(n) = (p-1)(q-1)$. It then selects a collection $\mathcal{P}$ of prime numbers. A scheme is used to pseudo-randomly select numbers sequentially from $\mathcal{P}$. The section is exclusive, i.e., a selected number will not be selected again in future's selections. An example of selection scheme is to pseudo-randomly permute the numbers in $\mathcal{P}$, and select numbers according to the resulting order. Let $p_i = \mathcal{P}[i]$ denote the *i-th* number of the numbers selected by the scheme from the collection $\mathcal{P}$ which are relatively prime to $\phi(n)$. In other words, $p_1, p_2, p_3, \cdots$ are the numbers selected sequentially from $\mathcal{P}$ by the scheme which are all relatively prime to $\phi(n)$, and no two numbers are equal. The number n and the collection $\mathcal{P}$ are publicly known, while the number $\phi(n)$ and the scheme to select numbers from $\mathcal{P}$ are secret information. The reason that the selection scheme is kept secret is to prevent a hacker from knowing the numbers selected from $\mathcal{P}$ by the scheme which are not relatively prime to $\phi(n)$ since such information can be used to deduce the secret $\phi(n)$. The pair of numbers $\{n, \phi(n)\}$ as well as the collection $\mathcal{P}$ and the selection scheme are shared with the license server which will use the information to derive content encryption keys during license acquisition. In an application setting, it is possible that the center in multicast also serves as the license server for the DRM system.

Multicast: Let B_i denote the *i-th* encryption block and k_i denote the corresponding encryption key in multicast. To generate the first content encryption key k_1, i.e., the encryption key for the first encryption block B_1, the center chooses a random integer $s \in (1, n)$, where s is relatively prime to n, and obtains the first prime number $p_1 = \mathcal{P}[1]$ from the collection $\mathcal{P}$. The first encryption key k_1 is set to be $s^{p_1^{-1}} \mod n$. The number s is also shared with the license server in order to generate keys to deliver in licenses to clients to replay saved contents. Note that s is not kept as a secret. It is publicly available.

When the first rekeying is needed, the center generates a new content encryption key k_2 to encrypt the second encryption block B_2. To do so, the center obtains the second prime number p_2 from $\mathcal{P}$, $p_2 = \mathcal{P}[2]$. The content encryption key k_2 for the second encryption block B_2 is given by $k_2 = s^{p_2^{-1}} \mod n$. This procedure is iterated whenever rekeying is needed. In general, for *i-th* rekeying, the center generates $(i+1)$-*th* content encryption key k_{i+1} to encrypt the $(i+1)$-*th* encryption block B_{i+1}. This is done by first obtaining the $(i+1)$-*th* prime number p_{i+1} from the collection $\mathcal{P}$: $p_{i+1} = \mathcal{P}[i+1]$. The $(i+1)$-*th* content encryption key k_{i+1} is given by $k_{i+1} = s^{p_{i+1}^{-1}} \mod n$, where $i \geq 0$. The prime number p_i is packaged into an auxiliary field in the encryption block B_i to be sent to clients. Alternatively, these prime numbers can be multicast in-band to clients separately from the content. In describing the scheme, we have assumed that there is no loss in transmitting the selected prime numbers to clients. If that is not the case in an application, error correction or redundant transmission of the selected prime numbers is needed to ensure that clients receive all the transmitted prime numbers. Note that those transmitted prime numbers are not used in multicast decryption. They will be used in generating required keys by the client side DRM module in decrypting saved content.

Recording: In this scheme, a client simply saves the encrypted content received from restricted multicast directly into the local storage for subsequent replays. There is no need to perform any transcryption. Suppose that a user wants to record $j+1$ consecutive encryption blocks from block B_i to block B_{i+j} for replays in future. The encryption blocks $B_i, \cdots, B_{i+j}$ along with the prime numbers $p_i, \cdots, p_{i+j}$ received from multicast are saved to the local storage. The saved files can be distributed to other users or devices if needed since, like in a conventional DRM system, the protected content can be redistributed without any restriction. Without a proper license, a user cannot play protected content.

A multicast program may contain some uninterested content such as advertisement that a user does not want to replay with the recorded file. In other words, a user may save inconsecutive encryption blocks into a file. A saved file containing inconsecutive encryption blocks can also occur when multiple saved files are merged into a single file. The integrated key scheme works with a saved file containing inconsecutive encryption blocks too. In fact, the scheme works equally well when a saved file consists of an arbitrary combination of encryption blocks $\{B_i | i \in I\}$, where I is an arbitrary set of block indices.

Consuming recorded content: A valid license with appropriate rights is required before a user can consume the recorded content. A client acquires a license from the license server. During license acquisition, the client sends to the license server the collection of block indices I that the saved file contains along with other identification information. A payment might be required during this process. After authenticating the client, the license server generates a secret key $s_I = s^{\pi_I^{-1}} \mod n$ for the encryption blocks $\{B_i | i \in I\}$ where $\pi_I \equiv \prod_{i \in I} p_i$, and delivers the key in a license to the client. Note that a single key is contained in the license to be delivered to a client.

In playing a recorded encryption block $B_i, i \in I$, the client side DRM module calculates the corresponding decryption keys[2] k_i from both the key s_I contained in the license and the received prime numbers with the equation: $k_i = s_I^{\pi_I / p_i} \mod n$. With the decryption key k_i, the client DRM module

[2]Symmetric encryption is assumed to be used in encrypting content. Therefore the decryption key is the same as the encryption key.

can decrypt the *i-th* block B_i. This process can be applied to all the recorded encryption blocks $\{B_i|i \in I\}$ contained in the saved file. As to be proved in the next section, the client can derive only the content encryption keys associated with the blocks $\{B_i|i \in I\}$, which is exactly the permission of decryption granted by the license server during the license acquisition. It will not be able to derive encryption keys associated with any other blocks. In other words, our integrated key scheme performs exactly as it is requested.

The above procedure applies only to the recorded content. It does not affect multicast playing. In fact, content in multicast can be played as usual as if our scheme did not exist.

IV. Security Analysis

Security of the integrated content encryption key scheme is analyzed in this section. We will argue that security of the proposed integrated key scheme depends on deduction of $\phi(n)$, which is usually done by factoring n. Factoring a large integer is believed to be a hard problem that security of the RSA encryption scheme relies on. We conclude in this section that the integrated key scheme is secure.

We first prove that the keys $\{k_i|i \in I\}$ can derive the key s_I and vice versa with the selected prime numbers available. From the previous section, key k_i is calculated by $s_I^{\pi_I/p_i} \mod n$. Reversely, we want to compute s_I from the keys $\{k_i|i \in I\}$. Since the integers $\{p_i|i \in I\}$ are different prime numbers, the integers $\{\pi_I/p_i|i \in I\}$ are coprime. Recall that $\pi_I \equiv \prod_{i\in I} p_i$. Therefore there exists a set of integers $\{\alpha_i|i \in I\}$ such that $\sum_{i\in I}(\pi_I/p_i)\cdot\alpha_i = 1$. The set of integers $\{\alpha_i|i \in I\}$ can be calculated with the Euclidean algorithm. Then we have $\prod_{i\in I} k_i^{\alpha_i} = s_I$, i.e., s_I can indeed be calculated from the keys $\{k_i|i \in I\}$. We conclude that knowledge of the keys $\{k_i|i \in I\}$ is equivalent to knowledge of the key s_I when all used prime numbers are publicly available.

Now security of the integrated scheme is equivalent to the following mathematical problem:

Given a message m, a positive integer $n = pq$ of the product of two large unknown primes p and q, and a set of signatures $c_i = m^{d_i} \mod n$ signed with the distinct RSA private keys $\{d_i|i > 0\}$, find the signature $c_0 = m^{d_0}$ signed by another private key d_0, where all the corresponding RSA public keys $\{e_i|e_i \cdot d_i = 1 \mod \phi(n), i \geq 0\}$ are distinct primes.

We believe that the above problem is a hard problem. Like other hard problems in cryptography, we cannot prove mathematically that the problem is indeed a hard problem. It seems to be difficult to transform the above problem into another problem widely believed to be hard. As a result, we cannot show rigorously that the integrated key scheme is secure. However, we are unaware of any efficient method to solve the above problem. A known method to solve the problem is still to deduce $\phi(n)$ by factoring n, which is believed to be a hard problem. We would argue that the integrated scheme should be secure. This problem will be left as an open problem to be solved in the future.

The public information of distributed prime numbers $\{p_i\}$ can be used to gain some knowledge of the secret $\phi(n)$, i.e., $\phi(n)$ cannot be divided by these numbers. If multicast has rekeyed $l-1$ times, the search space for $\phi(n)$ shrinks to $\prod_{i=1}^{l}(p_i - 1)/\prod_{i=1}^{l} p_i$ relative to the original search space. If all the prime numbers $\{p_i\}$ are large enough, and the total number of blocks is not very large, shrinking of the searching space is relatively small and should not cause any noticeable security vulnerability.

V. Discussion

The integrated content encryption key scheme described in Section III offers a few major advantages over the two straightforward methods described in Section II. Compared with the transcryption scheme, it has the simplicity that only a single key is generated and transmitted in a license to a client yet without the computational overhead, the complexity to negotiate a DRM encryption key with the license server for re-encryption, or the additional security requirements as the the transcryption scheme. No transcryption is needed in the integrated scheme. Content is saved directly to the local storage for subsequent replays. Compared with the direct recording method, the integrated scheme has the same efficiency to save the content directly yet without the burden of complex key management or large number of keys delivered in a license to a client. In other words, the integrated scheme combines the advantages of both straightforward methods.

The integrated scheme has an additional advantage over the the transcryption method when multiple recorded files containing consecutive or overlapping blocks need to be merged into a single one. In the transcription method, another transcryption is required since those recorded files are encrypted with different content encryption keys. Transcryption ensures that the merged content is encrypted with a single encryption key so that a license server knows how to generate the key when a client acquires a license. Merging can be easily done for both the integrated scheme and the direct recording method by simply concatenating those recorded files together according to the sequence number of the encryption blocks and removing the duplicated blocks. A new license is required to play the merged file. For the integrated scheme, the license server can easily generated a new key and deliver the single key to a client. For the direct recording method, all the encryption keys associated with the encryption blocks contained in the merged file have to be delivered to the client. This means that the size of a license increases.

The gain of the advantages of the integrated scheme comes at the cost of complexity in generating the content encryption keys by the multicast center, license server, and the client. The license server needs to do one scalar multiplication for issuing each license. Scalar multiplication is complex. The integrated scheme requires a collection of prime numbers. The prime numbers in the collection can be eventually exhausted for a long enough period of time of multicasting. A larger collection of prime numbers may alleviate the problem. The integrated scheme is suitable for those applications where rekeying occurs relatively infrequently so that the number of prime numbers in the collection would be large enough for the applications.

VI. Conclusion

In this paper we first described the challenges when DRM is applied to provide persistent managements of the rights for the content recorded from restricted broadcast or multicast. After presenting two straightforward solutions and discussing their drawbacks, we presented a novel integrated content encryption key scheme for broadcast and multicast encryption which facilitates DRM protection of the recorded content. The integrated scheme is efficient that broadcast or multicast content can be saved directly to local files, and a license server can efficiently manage and generate keys delivered to clients. Only a single key needs to be delivered in a license to a client. Security of the integrated scheme was analyzed. The advantages and disadvantage of the three schemes were also discussed. The proposed integrated scheme can be applied together with any broadcast or multicast encryption schemes.

References

[1] A. M. Eskicioglu, *A Key Transport Protocol for Conditional Access Systems*, Proc. SPIE Security & Watermarking of Multimedia Content III, vol. 4314, pp. 139-148, San Jose, CA, January 22-25, 2001.

[2] A. Fiat and M. Naor, *Broadcast Encryption*, Advances in Cryptology - Crypto93, Lecture Notes in Computer Science vol. 773, pp.480-491.

[3] C. Blundo, A. De Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung, *Perfectly-secure key distribution for dynamic conferences*, Advances in Cryptology C CRYPTO'92, Lecture Notes in Computer Science, vol. 740, pp. 471 - 486.

[4] M. Abdalla, Y. Shavitt, and A. Wool, *Key Management for Restricted Multicast Using Broadcast Encryption*, IEEE/ACM Trans. Networking, vol. 8, no. 4, pp. 443-454, 2000.

[5] R. Canetti, T. Malkin, and K. Nissim, *Efficient Communication Storage Tradeoffs for Multicast Encryption*, Advances in Cryptology C EUROCRYPT'99, Lecture Notes in Computer Science, 1999.

[6] I. Chang, R. Engel, D. Kandlur, D. Pendarakis, and D. Saha, *Key Management for Secure Internet Multicast Using Boolean Function Minimization Techniques*, IEEE INFOCOMM, 1999.

[7] J. Anzai, N. Matsuzaki and T. Matsumoto, *A Quick Group Key Distribution Scheme with Entity Revocation*, Advances in Cryptology C ASIACRYPT'99, Lecture Notes in Computer Science, 1999.

[8] R. Safavi-Naini and H. Wang, *New Constructions for Multicast Re-keying Schemes Using Perfect Hash Families*, Proceedings ACM Conference on Computer and Communication Security, CCS'00, Ahtens, Greece, 2000.

[9] J. Garay, J. Staddon and A. Wool, *Long-Lived Broadcast Encryption*, Advances in Cryptology - Crypto00, Lecture Notes in Computer Science 1880, pp.333-352.

[10] D. Halevi and A. Shamir, *The LSD Broadcast Encryption Scheme*, Advances in Crytology - Crypto02, Lecture Notes in Computer Science 2442, pp.47-60.

[11] P. DAroco and D.R. Stinson, *Fault Tolerant and Distributed Broadcast Encrytion*, CT - RSA03, Lecture Notes in Computer Science 2612, pp.263-280.

[12] R. Iannella, *Digital Rights Management (DRM) Architectures*, D-Lib Magazine, vol. 7, no. 6, June 2001.

[13] A. M. Eskicioglu, J. Town, and E. J. Delp, *Security of Digital Entertainment Content from Creation to Consumption*, Signal Processing: Image Communication, Special Issue on Image Security, vol. 18, no. 4, April 2003, pp. 237 - 262.

[14] Microsoft, *Microsoft Windows Media Digital Rights Management*, http://www.microsoft.com/windows/windowsmedia/drm/default.aspx.

[15] Open Mobile Alliance, *OMA DRM Specification Draft Version 2.0*, March 2004. http://www.openmobilealliance.org.

[16] *MPEG*, http://www.chiariglione.org/mpeg/

[17] *Digital Media Project*, http://www.dmpf.org/

[18] R. Rivest, A. Shamir, and L. Adleman, *A method for obtaining digital signatures and public-key cryptosystems*, Commun. ACM, vol. 21, no. 2, pp. 120-126, 1978.