Broadcast Encryption for Differently Privileged

Hongxia Jin[1] and Jeffery Lotspiech[2]

[1] IBM Almaden Research Center
San Jose, CA, 95120
jin@us.ibm.com
[2] Lotspiech.com, Henderson, Nevada
jeff@lotspiech.com

Abstract. Broadcast encryption is a primary technology that has been used for content protection. It enables a broadcaster to distribute content to a set of users so that only a privileged subset of users can access the content and another subset of revoked users cannot access the content. The main enabling block in a broadcast encryption scheme is the session key block, which each authorized user processes differently, but each gets the same valid session key. Currently all existing broadcast encryption schemes have assumed that the content and authorized users are equally privileged. There are emerging scenarios that demand protection for content with different privileges and for users with different privileges. In this paper we shall present a new broadcast encryption scheme that continues to employ single session key blocks but provides different privileged protections for different content and users. In particular we will expand the elegant subset-cover-based broadcast encryption scheme. We shall introduce a new concept called "security class" into the session key blocks. We use keys derived from a chain of one-way functions. Our approach is simple, efficient and secure.

1 Introduction

This paper is concerned with protection of copyrighted digital content. Piracy is becoming a more and more serious concern in the movie and music industries, since digital copies are perfect copies. Broadcast encryption is an important cryptographic key management approach, especially useful in content protection systems. It ensures content is distributed in a way that it is only accessible to a set of privileged (authorized) users and can exclude (revoke) another set of (compromised or non-compliant) users. In this paper, the devices that are used to decrypt and playback the content are interchangeably called devices or users. In a broadcast encryption system, each device is assigned a secret set of keys (called *device keys*). Another random chosen session key is indirectly used to encrypt the content. The content is usually video or music in a typical content protection system. The session key is also sometimes called the *media key*.

Device revocation is inherently tied to a broadcast encryption system. The fundamental enabling structure in a broadcast encryption for revocation is the session key block, or oftentimes called the media key blocks (MKB for short). It basically contains the media key encrypted by compliant device keys over and over. The MKB is distributed together with the encrypted content, for example, on the physical media.

D. Gritzalis and J. Lopez (Eds.): SEC 2009, IFIP AICT 297, pp. 283–293, 2009.
© IFIP International Federation for Information Processing 2009

During playback time, a device will always process MKB first. If a device is compliant, it can use one of its device keys to decrypt the MKB and obtain the valid media key which ultimately allows decryption of the content. The revoking devices will decrypt the MKB and get garbage, thus they cannot decrypt the content. Those devices are therefore called revoked by the MKB.

Two popular practical broadcast-encryption-based systems are the Content Protection for Recordable Media (CPRM) system from IBM, Intel, Panasonic, and Toshiba, and the Advanced Access Content System (AACS) from Disney, IBM, Intel, Microsoft, Panasonic, Sony, Toshiba, and Warner Bros.

As one can imagine, a media key block is naturally associated with a piece of content. All devices that are authorized have equal privilege to access that content. However, some recent emerging use scenarios demand protection for content with different values and for devices with different privileges. For example, in case where a consumer might have a library of entertainment content in his home, and wants that library to be freely viewed by all the devices he owns. As one can imagine, in this case, not all the content being protected is equally valuable. For example, the user might have some movies in standard definition and some movies in high definition. From the point of view of the movies' creators, the high definition version is more valuable, and would have more serious economic consequences if the users were to make unlimited unauthorized copies. Likewise, not all devices are equally privileged. There is no reason, for example, why a standard definition television needs a set of keys that allows it to decrypt high definition video. Furthermore, it is even possible that a single piece of content might contain different material that are of different values. For example, a high definition movie may come with some "coming attractions". These materials are of less value than the high definition movie itself. Therefore it is highly desirable to design a broadcast encryption scheme to enable protection for content with different values on devices with different privileges.

Of course, in order to do that, one solution is to employ multiple MKBs, one for each class of content. For protecting a piece of content containing materials that are of different value, this solution implies the complication of having multiple MKBs associated with one piece of content. To make it even worse, this solution does not always work. As will be shown in Section 2, to provide content protection in the above-mentioned home network scenarios, the devices do need a common media key block for other reasons and therefore simply employing multiple MKBs is not a feasible solution.

The main contribution of this paper is to expand the single media key block in the traditional broadcast encryption scheme design so that it can be used to protect content with different privileges as well as enabling devices to have different privileges. To achieve this goal, we will introduce different security classes into the traditional media key blocks and make use of hierarchical keys derived from a chain of one-way functions.

In rest of paper, in Section 2, we will use the home network as a real use scenario to clarify the context of our work. We will show why it is infeasible to employ multiple MKBs to provide different levels of protection in this scenario. Then in Section 3, we will show our design of a broadcast encryption scheme having a single media key block but in which all content is not equally protected and all devices are not equally privileged. In Section 4, we will formalize our new broadcast encryption scheme and prove its security.

2 Background on Content Protection for Home Network

In this paper we are motivated by protecting content within a home network. In a home entertainment network, several devices with various capabilities (eg., content storage and rendering) inter-operate across the network. Within a home network, all authorized devices form a cluster. Within a single home network, content may be freely moved and copied from device to device, because it remains encrypted and bound to those devices in the cluster. No restrictions are imposed on the transmission mechanism, the physical location or even the ownership of devices. Broadcast encryption technology can be used for secure home network. Readers refer to ASCCT content protection protocol used in HANA consortium, High Definition AudioVideo Network Alliance [2].

In a home network, the notion of *compliance* is important: devices must follow an agreed-upon set of rules regarding copyrighted content. A device not playing by the rules, i.e. a circumvention or non-compliant device, will be revoked. The objective of a secure home network is for all nodes in the cluster to compute a common key, so that each can verify that the others are compliant. (Non-compliant devices would be revoked in the media key block and would be unable to calculate the common key.) Therefore a common media key block is essential to securely form the cluster. In fact, the media key block now needs to be associated with a set of devices, not a particular item of content.

Furthermore, a common media key block is needed to enable the devices to remain in synchronization when new media key blocks revoke newly discovered circumvention devices. A cluster contains not just a common media key block, but also other data files, in particular the list of the authorized devices in the cluster. This authorization list must be cryptographically "signed" by the common key(s) in the cluster. Obviously, if there is more than one key in use in the cluster, synchronizing the signing when the new media key block is delivered is much more complicated. Thus, having multiple media key blocks, although theoretically possible, would greatly complicate the synchronization process.

Currently the ASCCT content protection protocol [2] uses one single media key block for the reasons above and in the protocol all devices are equally privileged. In order to expand it to enable differently privileged protection, one has to design a new broadcast encryption scheme that continues to employ a single media key block.

3 Protection for Content and Users with Different Privileges

In our approach we introduce the concept of "security class" into the media key blocks, corresponding to the different privileges of the devices in the system. Our approach retains a single media key block, with its straightforward synchronization, while still allowing different classes of devices to learn different keys from the same media key block. These different keys will allow devices in different security classes to access content with different privileges. The different keys come from a hierarchy of keys derived from a one-way cryptographic chain function. The chain of one-way functions allow a high security class device not only to access the content in its corresponding privilege but also to calculate the keys needed to access low privileged content. More importantly this also allows an easy synchronization cryptographically with the lower security class devices. That is the essence of our approach.

3.1 Generating a Single MKB That Enables Differently Privileged Devices to Access Differently Privileged Content

In order to expand a single media key block in a current broadcast encryption scheme to support multiple privileged devices to access multiple privileged content, we will first need to take a look at a general broadcast encryption scheme at an abstract level. In a general broadcast encryption scheme, the devices are organized into overlapping subsets; each subset is associated with a device key. Each device belongs to many different subsets and knows the key for every subset it is a member of. In order to create a MKB that can enable all compliant devices and exclude all non-compliant devices, the license agency will first find a subset cover that "covers" all innocent devices and exclude all compromised devices. Every subset is associated with a key. The media key block comprises encryption of the media key with each of the keys associated with the chosen subsets in the subset cover. To construct a minimal size MKB, one wants to find the minimal subset cover.

More formally, let $\mathscr{D}$ be the set of devices and $\mathscr{K}$ be the set of device keys. Every device $d \in \mathscr{D}$ owns a subset of keys, denoted by $\mathscr{K}_d$. Similarly, associated with every key $k \in \mathscr{K}$ is a set of devices $\mathscr{D}_k = \{d \in \mathscr{D} : k \in \mathscr{K}_d\}$.

Suppose we want to broadcast some media M, which, for all intent and purposes, is a binary string. We would like to encrypt M in such a way that a set of legitimate devices $L \subseteq \mathscr{D}$ is able to decrypt and view the media. The first step is to encrypt M with some key Km, referred to as the *media key*. We will use the term *key* without a qualifier to refer to device keys. We then find a subset of device keys C such that all legitimate devices are *covered*. That is, C is chosen such that $\bigcap_{k \in C} \mathscr{D}_k = L$. Now, for every $k \in C$ we separately encrypt the media key, giving us $E_k(Km)$. Ultimately, the following items are broadcasted.

- The encrypted media: $E_{Km}(M)$
- The encrypted media key (MKB): $\langle E_{k_1}(Km), E_{k_2}(Km), \ldots, E_{k_{|C|}}(Km) \rangle$
- An index of the device keys used to encrypt the media key: $\langle 1, 2, \ldots, |C| \rangle$.

Every device $d \in L$ will own a key used in the MKB and every device $r \in \mathscr{D}/L$, referred to a *revoking device*, will own none. Hence, it cannot recover the content.

To expand the media key block so that it can enable differently privileged devices to access differently privileged content, we introduce a concept called device "security class" into the media key block. Each security class corresponds to a different privilege. Traditional broadcast encryption schemes view all compliant devices belonging to one same security class. Our goal is to design a broadcast encryption system in which the authorized (compliant) devices belong to different security classes.

To do that, in our approach when we organize the devices into subsets, we will group devices in the same security class into same subsets as much as possible. We will show how to construct a MKB from a simple tree-based broadcast encryption scheme as shown in Figure 1. As show in [7], in a tree-based broadcast encryption scheme, all devices are organized to be the leaves of a tree. Each node, including internal nodes, is associated with a key. Each device is assigned, as its device keys, all the keys associated with the nodes on the path from the root to the leaf where the device sits. We also arrange all devices with the same security class in the same subtree. For example, device

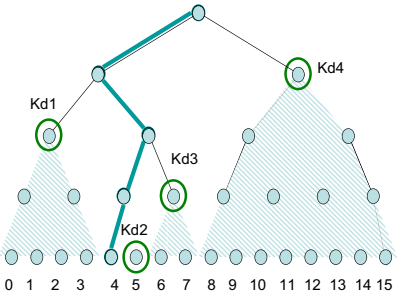

Fig. 1. A simple tree-based MKB

#0 to #3 belong to security class A. Device #4 to #7 belong to security class B. Device #8 to #15 belong to security class C.

In the example shown in Figure 1, device #4 is revoked. To distribute content that is accessible to every device except device #4, one needs to avoid encrypt media key Km using any key that device #4 knows, which is on the highlighted path in the figure. In order to construct a MKB, the first step is to find a subset cover C that covers all innocent devices. In this example, $C = \{K_{d1}, K_{d2}, K_{d3}, K_{d4}\}$. A traditional MKB contains

$$\langle E_{k_{d1}}(Km), E_{k_{d2}}(Km), E_{k_{d3}}(Km), E_{k_{d4}}(Km)\rangle.$$

For the example shown in Figure 1, in our approach, suppose there are three different security class devices in the system, each is authorized to play different classes of content. Suppose further class A content requires the least amount of security, class B content requires a higher level of security, and Class C requires the highest level of security. A device in security class A is configured to play only class A content. Devices in security class B is configured to play class A content and class B content. Devices in security class C is configured to play class A content, class B content and class C content C. For example, Class A content may be standard definition content, Class B may be high definition content, and Class C may be some premium content requiring even higher level of security than high-definition content.

In our approach, we will associate a common media key block (MKB) with a set of devices in different security classes so that they can access content with different privileges accordingly. This perfectly fits in the home network scenarios described earlier. A set of devices at one's home with different security classes form a cluster. Content with different privileges are allowed to move freely among all devices in the home cluster.

To construct a common MKB for a set of devices in different security classes, in our approach we will use multiple valid media keys, one for each security class. For the example above, there are three different media keys $Km1, Km2, Km3$ for the three different security class devices, $Km1$ for class A, $Km2$ for class B and $Km3$ for class C.

Furthermore, the valid media keys are not independently randomly chosen. In fact, only the media key for the highest security class is randomly chosen. The remaining media keys are derived from a chain of one-way function operations. In the above example, since $Km3$ is for the highest class C devices to access class C content, $Km3$ is randomly chosen. But $Km2 = f(Km3)$ and $Km1 = f(Km2)$ where $f()$ is a one-way function. Oftentimes we refer to the media key for the lowest security class simply as the *media key*, and we refer the media keys for the higher security classes as *media key precursors*. So $Km1$ is a media key, $Km2, Km3$ are media key precursors.

When constructing a common MKB in our approach, the different subset keys in the cover C will now encrypt different media keys instead of encrypting one same media key. In fact for the above example shown in Figure 1, in our approach the following items are broadcasted.

- The encrypted media: $E_{Km1}(M1)$, $E_{Km2}(M2)$, $E_{Km3}(M3)$.
- The encrypted media keys (common MKB): $\langle E_{k_{d1}}(Km1), E_{k_{d2}}(Km2), E_{k_{d3}}(Km2), E_{k_{d4}}(Km3)\rangle$.
- An index of the device keys used to encrypt the media keys

When a class A device extracts the common MKB and uses its device key to process the MKB, it will calculate a media key $Km1$ allowing decryption of only Class A content. When a class B device process the same MKB, it calculates a media key precursor $Km2$ which allows decryption of class B content. Class B devices also have the ability to process class A content. To do that, it will use the media key precursor $Km2$ and a one-way function to calculate a media key $Km1$ to decrypt class A content.

Similarly, when a class C device processes the common MKB, it will calculate a media key precursor $Km3$ which allows decryption of class C content. It may also decrypt class B content by calculating a media key precursor $Km2$ from the $Km3$ using the one-way function. This media key precursor $Km2$ may be used to decrypt class B content. Likewise, media player C may also process class A content by calculating a media key $Km1$ from the media key precursor $Km2$ using the one-way function. This media key $Km1$ may be used to decrypt class A content.

Our approach may utilize a variety of known one-way functions. For example, the following well-known one-way function, based on the Advanced Encryption Standard (AES) cipher, can be used:

$$r = AES\_D(k,d) \oplus d$$

where r is the result, k is a key, d is data, AES_D is AES decryption in electronic code book mode. This function is one-way in the following sense: from r, it is intractable to calculate either k or d, even if one of the values is known.

In our use, k would be a media key precursor and d would be a constant known to all devices. Note that d does not have to be a secret. It can be a published constant without hurting the security of our approach.

3.2 Protect Differently Privileged Materials within Content

In practice, media key is rarely directly used to encrypt content. There is often at least one level of indirection: the content is encrypted with a key, called *title key*; and then

the title key is encrypted with the media key from the media key block. This encrypted title key is typically stored in a header associated with the content. As a result, for the example in Figure 1 the following items are actually broadcasted:

- The encrypted media and *title keys*: $E_{Km1}(Kt1), E_{Km2}(Kt2), E_{Km3}(Kt3); E_{Kt1}(M1), E_{Kt2}(M2), E_{Kt3}(M3)$.
- The encrypted media keys (common MKB): $\langle E_{k_{d1}}(Km1), E_{k_{d2}}(Km2), E_{k_{d3}}(Km2), E_{k_{d4}}(Km3)\rangle$.
- An index of the device keys used to encrypt the media keys

If a piece of content contains material that are of different value, it is also possible to use our approach to provide different privileged protection. Those materials with different privileges will be encrypted with different title keys. And the media keys for different security classes derived from the one-way chain can be used to encrypt the different title keys. For example, for a high definition movie that also contains lower valued "coming attractions", different title keys are used to encrypt the high definition movie content and the "coming attractions". One can use media key precursor to encrypt the title key for the high definition movie and use the media key to encrypt the title key for the "coming attractions".

As we can see, we have provided a general broadcast encryption system that employs a common media key block, while providing different levels of protection for different media and different devices, be it for different content with different privileges or for materials with different privileges within the same piece of content.

4 Formalization and Security Proof

Our scheme is expanded from a traditional single security class broadcast encryption scheme to multiple security classes. In our newly expanded scheme, the device key assignment can be exactly same as that in a traditional broadcast encryption scheme. In particular we expand the elegant state-of-art subset cover based NNL scheme [7] to enable multiple security classes. Our expanded scheme consists of the following algorithms:

1. *Setup*: Let $\mathscr{D}$ be the set of devices and $\mathscr{K}$ be the set of device keys. Every device $d \in \mathscr{D}$ is assigned a subset of keys, denoted by $\mathscr{K}_d$, based on subset cover NNL scheme key assignment. For device d, all the secret information including its device keys is denoted I_d.
2. *Subset cover:* Associated with every key $k \in \mathscr{K}$ is a set of users $\mathscr{D}_k = \{d \in \mathscr{D} : k \in \mathscr{K}_d\}$. Given a set of legitimate devices $L \subseteq \mathscr{D}$, find the subset cover of device keys C such that all legitimate devices are *covered*. That is, C is chosen such that $\bigcap_{k \in C} \mathscr{D}_k = L$.
3. *Encryption(M_j): encrypting a message belong to security class j*: Suppose there are s security classes. We will use s different media keys. $Km_{i+1} = f(Km_i)$. Suppose C contains m subsets which will be distributed among s security classes. There are m_j subsets in security class j and $\sum_{j=1}^{s} m_j = m$. Assume subsets $i_{j,1}, i_{j,2}, \cdots, i_{j,m_j}$ belong to security class j. Let $k_{j,1}, k_{j,2}, \cdots, k_{j,m_j}$ be their corresponding keys. Encrypt each content M_j with $E_{Km_j}(M_j)$ and put

$E_{k_{j,1}}(Km_j), E_{k_{j,2}}(Km_j), \cdots, E_{k_{j,m_j}}(Km_j)$ into MKB. The encryption methods $E(M)$ and $E(Km)$ can also be the same ones as those in NNL scheme.

4. *Decryption(j)*: For a device in security class j, use its device key $k_{j,i}$ to decrypt its corresponding part in MKB to get Km_j. Use Km_j to decrypt the encrypted content $E_{Km_j}(M_j)$ to get M_j.

Below is the general comparison. In a traditional broadcast encryption scheme the following items are actually broadcasted:

$$\langle E_{Km}(M), [E_{k_1}(Km), E_{k_2}(Km), \ldots, E_{k_{|C|}}(Km); 1, 2, \ldots, |C|]\rangle$$

In our s-security class broadcast encryption scheme we will distributes the s security class content messages containing the following items:

$$\begin{aligned}
&\langle E_{Km_1}(M_1), E_{Km_2}(M_2), \cdots, E_{Km_s}(M_s);\\
&[E_{k_{1,1}}(Km_1), E_{k_{1,2}}(Km_1), \cdots, E_{k_{1,m_1}}(Km_1),\\
&\ E_{k_{2,1}}(Km_2), E_{k_{2,2}}(Km_2), \cdots, E_{k_{2,m_2}}(Km_2),\\
&\ \cdots,\\
&\ E_{k_{s,1}}(Km_s), E_{k_{s,2}}(Km_s), \cdots, E_{k_{s,m_s}}(Km_s),\\
&\ i_{1,1}, i_{1,2}, \cdots, i_{1,m_1},\\
&\ i_{2,1}, i_{2,2}, \cdots, i_{2,m_2},\\
&\ \cdots,\\
&\ i_{s,1}, i_{s,2}, \cdots, i_{s,m_s},]\rangle
\end{aligned}$$

Our $E(M)$ and $E(Km)$ methods must satisfy the same property as those in NNL scheme. For example the method $E(Km)$ in our scheme has to be CCA-1 secure in the following sense: consider a feasible adversary $\mathscr{B}$that for a random key Kd gets to adaptively choose polynomially many inputs and examine E_{Kd}'s encryption and similarly provide ciphertext and examine E_{Kd}'s decryption. Then $\mathscr{B}$is faced with the following challenge: for a random plaintext message M, it gets back $E_{Kd}(M')$ where M' is either equal to M (or the 'real' case), or is equal to a totally random message (or the random case). The encryption method $E(Km)$ is CCA-1 secure if no polynomial adversary can distinguish these two cases with non-negligible advantage.

We used the same device key assignment in our expanded scheme in the setup step as NNL scheme, therefore our scheme shares the same *key indistinguishability* property. That is, for every subset S_i its associated key Kd_i is indistinguishable from a random key given all the information of all users that are not in S_i.

4.1 Definitions

Now we will prove the semantic security of our broadcast encryption revocation scheme. Intuitively, it more or less states that only the users in the designated security class and *its ancestors* can decrypt messages that are sent to that security class users, while no other users of the system can. The other users include the revoked users and the users in lower security class. By ancestor, we mean the users in a higher security class.

Formally we will define the first CCA-1 security of our broadcast encryption revocation scheme as follows:

Definition 1. *Consider an adversary $\mathscr{B}$ that gets to*

1. *Queries. Selects adaptively a set R of users and obtain I_u for all $u \in R$. By adaptively we mean that $\mathscr{B}$ may select messages $\mathscr{M}_1, \mathscr{M}_2, \cdots$ and revocation set $R_1, R_2, \cdots$. Each message can be divided into s smaller messages. In other words, $\mathscr{M}_i = \{m_{i1}||m_{i2}||\cdots||m_{is}\}$). For each class j the center randomly choose a session key Km_j. $\mathscr{B}$ queries center with $< m_{ij}, j >$ and R_i for each $1 <= j <= s$. The center returns the encryption of each $m_{ij}, 1 <= j <= s$ with Km_j when the revoked set is R_i. Also $\mathscr{B}$ can create a ciphertext for a class j and see how any (non-corrupted) user in class j decrypts it. It then asks to corrupt a receiver u and obtains its I_u. This step is repeated $|R|$ times (for any $u \in R$).*
2. *Challenge. Choose a message $M = \{m_1||m_2||\cdots||m_s\}$ as the challenge plaintext and a set R of revoked users that must include all the ones it corrupted (but may contain more). For each m_j, choose a random message Rm_j of similar length. For each j, ask the center to use security class j keys to encrypt message m_j or RM_j.*

$\mathscr{B}$ then receives all encrypted messages $M' = \{m'_1, m'_2, \cdots, m'_s\}$ with a revoked set R. For each j, it has to guess whether $m'_j = m_j$ or $m'_j = Rm_j$. We say a s-security class revocation scheme is secure if no polynomial adversary can distinguish between these two cases with non-negligible advantage.

4.2 The Security Theorem

It is not hard to imagine that the security of the *s*-security class revocation scheme depends on the traditional 1-security class revocation scheme which further depends on the device key indistinguishability. In the following, we state the security theorem and prove the *s*-security class revocation scheme has the same security strength as the 1-security class scheme under the same setup parameters.

Theorem 1. *Let $\mathscr{A}$ be an adversary that distinguishes ciphertexts defined in Definition 1 against our s-security class subset cover revocation scheme, and succeeds with probability δ in time τ. Then there exists an algorithm $\mathscr{B}$ which breaks the 1-security class subset cover revocation scheme with success probability $\delta' \approx \delta$ in time $\tau' = \tau$.*

Proof. The setup step is same for both schemes with same parameters. Suppose there is an adversary $\mathscr{A}$ against the *s*-security class subset-cover revocation scheme with success probability δ. Then we can construct an algorithm $\mathscr{B}$ that uses $\mathscr{A}$ as a subroutine to break the 1-security class subset-cover revocation scheme with a probability of $\delta' \approx \delta$.

Consider an adversary $\mathscr{B}$ that gets to

1. *Queries.* Select adaptively a set R of users and obtain I_u for all $u \in R$. By adaptively we mean that $\mathscr{B}$ may select adaptively messages $\mathscr{M}_1, \mathscr{M}_2, \cdots$ and revocation set $R_1, R_2, \cdots$. $\mathscr{B}$ divides each message into s smaller messages. In other words, $\mathscr{M}_i = \{m_{i1}||m_{i2}||\cdots||m_{is}\}$. $\mathscr{B}$ forwards a query with $\mathscr{M}_i$ and R_i to adversary $\mathscr{A}$. $\mathscr{A}$ randomly chooses a class $t, (1 <= t <= s)$ and asks the center to encrypt

each sub-message m_{ij} with class t key. The center returns the encryption of each $m_{ij}, 1 <= j <= s$ with security class t key Km_t when the revoked set is R_i. $\mathscr{A}$ aggregates the returned message together and forwards to $\mathscr{B}$. Adversary $\mathscr{B}$ can also create a ciphertext and forward it to $\mathscr{A}$ to see how any (non-corrupted) user in class t decrypts it. Whenever $\mathscr{B}$ decides to corrupt a user u, $\mathscr{A}$ asks I_u from the center and replies I_u to $\mathscr{B}$. The above step is repeated $|R|$ times (for any $u \in R$).

2. *Challenge.* $\mathscr{B}$ chooses a message $\mathscr{M} = \{m_1||m_2||\cdots||m_s\}$ as the challenge plaintext and a set R of revoked users that must include all the ones it corrupted in the query phase as well. Adversary $\mathscr{B}$ forwards the same challenge plain text with the same R to $\mathscr{A}$. Choose a random message Rm of similar length with M and divide into $Rm = \{Rm_1||Rm_2||\cdots||Rm_s\}$, each Rm_j is of similar length with m_j. Asks the center to use security class t key to encrypt message m_j or Rm_j.

$\mathscr{A}$ receives encrypted messages with the revoked set R and aggregate together $M' == \{m'_1, m'_2, \cdots, m'_s\}$ to return to adversary $\mathscr{B}$. Adversary $\mathscr{B}$ has to guess whether $m'_j = m_j$ or $m'_j = Rm_j$. $\mathscr{A}$ does the guess and forwards its answer back to $\mathscr{B}$. $\mathscr{B}$ will use $\mathscr{A}$'s answer as its answer.

If $\mathscr{A}$ succeeds with probability of δ in distinguishing the two cases, $\mathscr{B}$ succeeds with the same probability $\delta' \approx \delta$ and running time of algorithm $\mathscr{B}$ is same as that of $\mathscr{A}$ś.

The above theorem shows that revoked users cannot access the encrypted content sent to any authorized security class users. We believe it is also possible to show that, if media key $Km_i = f(Km_j)$ where $j < i$, then any non-revoked users in security class i cannot access content for security class j while the reverse is true. It is straightforward to see that this security relies on the intractability of the one-way function and the CCA-1 secure property of the encryption method $E(Km)$. More formally, an adversary cannot win the following game with non-negligible advantage.

Let $Km_1, \cdots, Km_s$ be the s media keys derived from the one-way function chain such that $Km_{j+1} = f(Km_j)$. Consider a feasible adversary $\mathscr{B}$ that

1. Selects j, $j \in [1, s]$, the adversary is allowed to access any key that is derivable from Km_j, but not Km_j and its ancestor.
2. *Queries*: Select any key Km_i where $j < i$, in other words, Km_i is derivable from Km_j. Adversary $\mathscr{B}$ gets to adaptively select polynomially many inputs and examine E_{Km_i}'s encryption and similarly provide ciphertext and examine E_{Km_i}'s decryption.
3. *challenge*: The adversary chooses a random plaintext message M, it gets back $E_{Km_j}(M')$ where M' is either equal to M (or the 'real' case), or is equal to a totally random message (or the random case). The adversary $\mathscr{B}$ cannot distinguish the two cases with non-negligible advantage.

5 Conclusion

In traditional broadcast encryption schemes every authorized user has equal privilege to access content. There is emerging need to enable differently privileged users to access differently privileged content. In this paper we have presented a new broadcast encryption scheme that can achieve this goal. In particular we have expanded the elegant

subset-cover-based broadcast encryption scheme. We introduced a new concept called "security class" into the session key blocks. We use session keys derived from a chain of one-way functions. Each session key corresponds to one security class. This avoid using multiple session key blocks but still achieve our goal. Our approach is simple, flexible, efficient and secure.

References

1. http://www.aacsla.com/specifications
2. http://www.hanaalliance.org/
3. Fiat, A., Naor, M.: Broadcast encryption. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 480–491. Springer, Heidelberg (1994)
4. Chor, B., Fiat, A., Naor, M.: Tracing traitors. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 257–270. Springer, Heidelberg (1994)
5. Chor, B., Fiat, A., Naor, M., Pinkas, B.: Tracing traitors. IEEE Transactions on Information Theory 46, 893–910 (2000)
6. Naor, M., Pinkas, B.: Efficient trace and revoke schemes. In: Frankel, Y. (ed.) FC 2000. LNCS, vol. 1962, pp. 1–20. Springer, Heidelberg (2001)
7. Naor, D., Naor, M., Lotspiech, J.: Revocation and tracing schemes for stateless receivers. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 41–62. Springer, Heidelberg (2001)
8. Boneh, D., Waters, B.: A collusion resistant broadcast, trace and revoke system. ACM Communication and Computer Security (2006)
9. Fiat, A., Tassa, T.: Dynamic traitor tracing. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 354–371. Springer, Heidelberg (1999)
10. Safani-Naini, R., Wang, Y.: Sequential traitor tracing. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 316–332. Springer, Heidelberg (2000)
11. Jin, H., Lotspiech, J., Nusser, S.: Traitor tracing for prerecorded and recordable media. In: ACM DRM workshop (October 2004)
12. Jin, H., Lotspiech, J.: Renewable traitor tracing: A trace-revoke-trace system for anonymous attack. In: Biskup, J., López, J. (eds.) ESORICS 2007. LNCS, vol. 4734, pp. 563–577. Springer, Heidelberg (2007)
13. Jin, H., Lotspiech, J., Megiddo, N.: Efficient Coalition Detection in Traitor Tracing. In: IFIP 23rd Information Security conference, Milan, Italy, pp. 365–380.