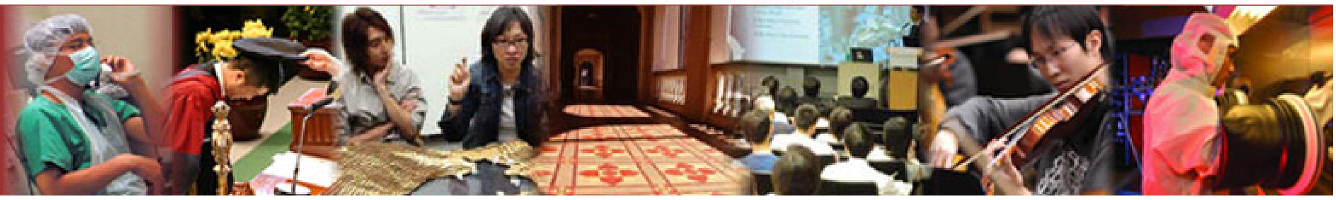

Avoid Illegal Encrypted DRM Content Sharing with Non-transferable Re-encryption

Yi-Jun He
Department of Computer Science
The University of Hong Kong
Chow Yei Ching Building
Pokfulam Road, Hong Kong
Email: yjhe@cs.hku.hk

Lucas C.K. Hui
Department of Computer Science
The University of Hong Kong
Chow Yei Ching Building
Pokfulam Road, Hong Kong
Email: hui@cs.hku.hk

Siu Ming Yiu
Department of Computer Science
The University of Hong Kong
Chow Yei Ching Building
Pokfulam Road, Hong Kong
Email: smyiu@cs.hku.hk

Abstract—**Digital rights management (DRM) technology enables valuable electronic media content distribution while preserving content providers' rights and revenues. Traditional DRM system utilizes security techniques to restrict copying of media content or allow only a single copy to be made. However consumers are demanding for the right to make copies for personal use or the right to use content on any device. Several DRM infrastructures have been proposed for secure content sharing. These infrastructures usually require cooperation and participation of both DRM technology providers and content providers; however there is a popular flaw in these schemes: the malicious employees of DRM technology providers can distribute DRM enabled contents to any consumers or make copies of a purchased content accessible to any devices without letting content provider know, thus reducing content providers' benefit.**

In this paper, we propose a novel DRM infrastructure which is based on a non-transferable re-encryption scheme to solve the above problem inherent in existing DRM infrastructures. In the proposed infrastructure, DRM technology providers and content providers are required to cooperate to make a purchased digital content for a specific device accessible by other different devices, and get extra profit from providing such services. The system preserves DRM technology providers and content providers' security properties while achieving secure and mutual profitable DRM content sharing. Furthermore, we allow content providers to trace the content, and control the content sharing rights. Even when malicious employees in DRM technology providers and DRM agent collude, they cannot re-delegate access rights to any device without permission from content provider, thus preserving content provider's benefit.

I. Introduction

Digital rights management (DRM) [1] is a term for access control technologies that are used by hardware manufacturers, publishers, copyright holders and individuals to limit the use of digital content and devices. A DRM system provides technologies, and defines rules to inhibit uses of digital content that is not desired or intended by the content provider. Usually, DRM is enforced using a protected content file and a license file. The license file will contain the usage rules, which are signed by a trusted authority and the protected content file will contain the encrypted content, which can be rendered only by devices possessing the corresponding license file. The devices will verify the signature of the license, verify the hash of the content, decrypt the content, and then be able to play the music, show the video, or run the game, depending on the type of content.

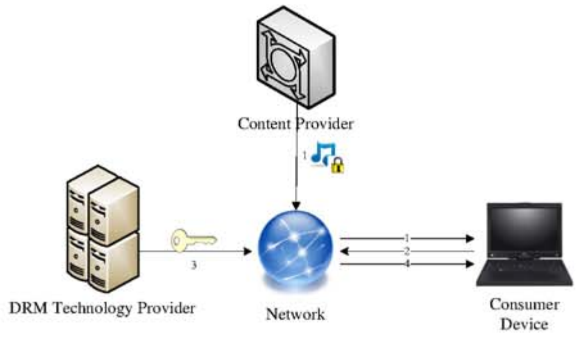

Fig. 1. The common process in DRM system infrastructure.

Different DRM systems have different DRM implementations and infrastructures, however, the basic DRM process is the same, which usually involves three parties: Content provider, DRM technology provider, Consumer. Figure 1 illustrates a simplified diagram of DRM content distribution from Content provider to a consumer.

1) The consumer gets the encrypted content using a download mechanism from network.
2) The consumer attempts to use the encrypted content by sending a request to the DRM thehnology provider through Internet for the license.
3) The DRM technology provider determines the policies based on the request. A financial transaction may be conducted for issuing the license.
4) The license is packaged and transferred to the client through Internet. Then the content is used on consumer device side corresponding to the request.

Traditional goal for a DRM is to prevent an end-user to make an unauthorized use of a piece of content (usually music or video). One example of DRM protection is DRM content purchased form a specific store may require specific devices for playing. Thus a direct copy of DRM content from a device may not be playable on another device. This DRM technology prevents illegal digital content sharing, and protects content

978-1-61284-307-0/11/$26.00 ©2011 IEEE

providers' benefit. Such DRM technology may have pleased content providers in the beginning, but with the wide usage of smart phones, MP3 players, MP4 players and other audio devices *et al.*, consumers require their purchased valuable digital content, such as music, books, videos, ring tones to be also shareable in those devices. However, traditional DRM techonolies hinder digital items coping and sharing. As a result, confused or dissatisfied customers can cause future customers to avoid legitimate digital content providers, and therefore, slows the growth of the digital industry. A recent survey by INDICARE [2] showed that consumers are willing to pay a higher price for more usage rights and device interoperability. From the web users polled, 86% preferred paying 1 Euro for a song that runs on any device rather than only 50 cents for a song that runs on only one device.

Many works [3], [7], [8], [10], [12] have been focused on the DRM content sharing problem. In general, the key solution for this problem is to translate the DRM content from one format to other formats accessible by other devices. Some solutions [3], [7] require a trusted third party to manage content translation, but these solutions rely too much on the third party. Once the third party is compromised or lost, anyone can use it to translate the content. Some researchers [8], [12] proposed to use proxy re-encryption schemes (PRE) for content sharing. However, we find that all these re-encryption based content sharing schemes cannot resist to a kind of collusion attack, which is described in detail in section III.

CONTRIBUTIONS.

We propose a new DRM infrastructure based on non-transferable proxy re-encryption scheme to tackle the DRM content sharing problem. More importantly, we provide better payment control to content sharing for preserving content provider's benefit. This new infrastructure inherits advantages of existing proxy re-encryption (PRE) based DRM infrastructure, which are

- Minimizing the provider's trust in the DRM agent during content and license translation by disallowing it access to the unprotected content (unless the DRM agent can break the underlying cryptography).
- Even if a DRM agent is compromised, it will not reveal the protected content.

Moreover, the new infrastructure can better protect content providers' benefit in terms of:

- Even if DRM agent colludes with malicious employees in DRM technology provider, they would not be able to conduct illegal DRM content sharing.
- Both content provider and DRM technology provider can get extra payment from providing DRM content sharing service, but neither of them can provide such service if working alone.

In Section 2 we introduce the existing research works and their disadvantages. In section 3, we briefly review the PRE scheme on which our scheme is based. Section 4 presents the system architecture and analysis while Section 5 discusses implementation and reports experimental results. Finally, Section 6 concludes the paper and outlines some future work.

II. Vulnerabilities of Previous DRM Schemes

Kravita *et al.* [7] suggested to use an external trusted party to manage content sharing. However, the external party will get to know all the security properties of the DRM technology providers. For privacy and commercial concern, providers are reluctant to share their security properties with external party to avoid malicious attack. For example, in 2005, the digital rights management (DRM) of Apple's iTunes was compromised partially due to the fact that an untrusted party (i.e., the client's resource) could obtain the plaintext during a naive decrypt-and-encrypt operation, albeit with symmetric encryption [6].

SmartPro [3] is a smart card based content protection system. Smart card (e.g., SIM card) securely stores system keys and guarantees the integrity of the software using these keys. The DRM system in a card will keep all private or shared symmetric keys secret - even from the card's owner. User who possesses the cards can access the content. It can also transfer content from a source domain to one or several other destination domains. The drawback of such method is that it relies too much on a hardware token, once the card is lost or stolen, the user will lose the content.

Nam *et al.* [10] proposed a method of using a neural format for content translation to achieve content sharing between different devices, which means that every DRM system shares the same security infrastructure. In their scheme, devices translate content to a neutral format when exporting it and then convert the received neutral format to their own DRM format while importing it. However, security of this scheme relies much on the device, because content translation and license generation are performed by device. Once the device is compromised or lost, anyone can use it to translate the content.

A new idea of using a secure proxy re-encryption (PRE) scheme to achieve secure content sharing is proposed in [8], [12]. A semi-trusted Domain Interoperability Manager (DIM) is introduced in these PRE based schemes, such that the DIM can perform the encrypted content translation. Taban *et al.*[12] presented two protocols. The first protocol minimized the provider's trust in the DIM during content and license translation by disallowing it access to the unprotected content. However, this protocol is not flexible because it requires strong assumptions about the exporting and importing devices and DRM systems. Thus the second protocol is proposed to extend the first one to a more flexible setting. However, the semi-trusted DIM can access the content by decrypting the content directly or decrypting the license to get the access right indirectly.

Lee *et al.* [8] allows a content provider to designate a DRM Agent to perform content translation, and claims to be able to achieve mutual profitable, which means providers can request additional fees for providing content sharing services. However, we find this scheme is not secure under a collusion attack: Malicious employees in DRM technology provider (also called DRM server in [8]) and DRM Agent can collude to make copies of a purchased content accessible to any devices without letting content provider know, thus an illegal content sharing could be done bypass content provider. This attack reduces content providers' benefit. The attack is described in detail in section III.

III. An Attack to Lee's DRM System

In this attack, after getting a re-encryption key to do re-encryption for only once, the designated DRM Agent (*DIA*) and malicious employees in DRM server (*DS*) can collude to generate another "re-encryption key" $g^{\pi/\mu}$ to decrypt any interoperable format cipher IC_m to get the content encryption key K_m without asking content provider (*CP*) for a re-encryption key any more. As a result, *DIA* and malicious employees in *DS* can take *CP*'s profit by selling the "re-encryption key". For instance, if a consumer wants to have a purchased DRM protected content *m* which is stored in *DIA* played in another device, under normal circumstances, *CP*, *DS* and DRM Interoperability Server (*DIS*) cooperate to provide the content sharing service, and get payment respectively. However, if the attack succeed, consumer just needs to pay *DIA* and *DS* for a re-encryption key $g^{\pi/\mu}$. But *CP* and *DIS* do not know the illegal trading among *DIA*, *DS* and consumer. The detailed attack procedure is like this:

After a successful content sharing, *DIA* gets hold of a re-encryption key $rk_{\mu\to\alpha} = g^{\alpha/\mu}$. This is a legal re-encryption key created by *DIS*, *DS* and *CP*. The detailed process of generating $g^{\alpha/\mu}$ can be found in [8]. In order to do an illegal content sharing bypass *CP* and *DIS*, malicious employees in *DS* and *DIA* try to generate an illegal re-encryption key from this legal re-encryption key. At first, *DIA* computes $g^{\alpha\pi/\mu}$ from $rk_{\mu\to\alpha}$. Then *DIA* gives $g^{\alpha\pi/\mu}$ to *DS*. *DS* computes $g^{\pi/\mu}$ using its private key α. $g^{\pi/\mu}$ is the illegal re-encryption key. If *DIA* and *DS* disclose this $g^{\pi/\mu}$ to any consumer *i*, *i* can decrypt the IC_m by computing

$$\varphi_2(K_m) = \varphi_2(K_m)Z^{\pi k_1}\Big/ e(g^{\pi/\mu}, g^{\mu k_1})$$

After getting $\varphi_2(K_m)$, consumer can decrypt *m* from $SE(K_m; m)$.

Note that this attack is not only successful to Lee's system, but also workable to existing proxy re-encryption based DRM systems because of the inherent weakness of proxy re-encryption schemes.

IV. Preliminary concepts

A. Proxy Re-encryption

A proxy re-encryption (PRE) scheme allows a proxy to re-encrypt a ciphertext for Alice (delegator) to a ciphertext for Bob (delegatee) without seeing the underlying plaintext. With the help of the proxy, Alice can delegate the decryption right to any delegatee. The Non-Transferable PRE scheme used in this paper were proposed in [4], [5]. We briefly recall Non-Transferable PRE scheme and some security properties here. For comprehensive definitions, please see the full version of the paper [4], [5]. The scheme is composed of the following algorithms:

- **Setup**. On input a security parameter 1^k, the public parameters $mpk = (g, g_1, h_1, h_2, h_3, H_I, H, H', \mathcal{M})$ and master secret key $msk = (\alpha)$ of PKG are generated. H_I, H and H' are secure hash functions. h_1, h_2, h_3, $g \in G$ and $\alpha \in Z_p$. It sets $g_1 = g^\alpha$. Define the message space $\mathcal{M} \in G_T$. We say that G_T has an admissible bilinear map *e*: $G \times G \to G_T$.
- **Key Generation**. User *A*'s private key is usk_A =(r_A, $r_{A,1}$, $h_{A,1}$, $r_{A,2}$, $h_{A,2}$, $r_{A,3}$, $h_{A,3}$), in which r_A, $r_{A,1}$, $r_{A,2}$, $r_{A,3}$ $\in$ Z_p, $h_{A,1} = (h_1 g^{-r_{A,1}})^{1/(\alpha - id_A)}$, $h_{A,2}$=$(h_2 g^{-r_{A,2}})^{1/(\alpha - id_A)}$, $h_{A,3}$=$(h_3 g^{-r_{A,3}})^{1/(\alpha - id_A)}$. Similarly, user *B*'s private key is denoted as $usk_B = (r_B, r_{B,1}, h_{B,1}, r_{B,2}, h_{B,2}, r_{B,3}, h_{B,3})$. *A* publishes her public key upk_A=$(p_{A,1}, p_{A,2})$, where $p_{A,1}$=${g_1}^{r_A}$, and $p_{A,2}$=$g^{r_A id_A}$.

- **Encryption**. The encryption algorithm $AE(upk, m)$ takes public key upk_A of delegator *A*, a unique randomly-selected secret parameter $s \in Z_p$, and message *m* as input, computes the ciphertext *C* where: C=(C_1,C_2,C_3,C_4,C_5,C_6) =(${p_{A,1}}^s {p_{A,2}}^{-s}$, $e(g, g)^s$, $m \cdot e(g, h_1)^{-s}$, $e(g, g)^{H'(m)}$, $g^{s\beta + H'(m)}$, $e(g, h_2)^s e(g, h_3)^{s\beta}$). We set $\beta = H(C_1, C_2, C_3, C_4)$.

- **Decryption(delegator)**. To decrypt a ciphertext $C = (C_1, C_2, C_3, C_4, C_5, C_6)$ using secret key usk_A, delegator Alice computes $\beta = H(C_1, C_2, C_3, C_4)$ and tests whether
$$e(C_5, g) = C_2^\beta C_4$$
and
$$C_6 = e(C_1, h_{A,2}{h_{A,3}}^\beta)^{1/r_A} \cdot {C_2}^{r_{A,2} + r_{A,3}\beta}$$
If either of them is not equal, outputs $\perp$. Else computes
$$m = C_3 \cdot e\left(C_1, h_{A,1}\right)^{1/r_A} \cdot {C_2}^{r_{A,1}}$$
If $e(g, g)^{H'(m)} = C_4$ holds, return *m*; otherwise return $\perp$.
- **Re-Encryption Key Generation**. *A* generates a random value $a_i \in Z_p$, where $i \geq 1$. a_i will be invalid after a time period *i*. *A* signs *B*'s identity ID_B, and sends the signature σ, ID_B, a_i to PKG via a secure channel. PKG verifies the delegator *A*'s signature, and extracts delegatee *B*'s *ID* from signature. The re-encryption key generation algorithm outputs a re-encryption key $rk_{A\to B}$=$(\frac{\alpha - id_B}{\alpha - id_A} + a_i y) \bmod p$, where *y* is a random number chosen by PKG. B_1=$\left(h_1{}^{r_B} g^{-r'_B}\right)^{a_i y/(\alpha - id_B)}$ is also computed for *A* to generate a Partial-Decryption-Key later.
- **Partial-Decryption-Key Generation**. *A* checks the correctness of the re-encryption key, and generates a partial decryption key (${h'_B}^{1/r_A}$, ${B_1}^{1/r_A}$), where h'_B is from *B*. Then the partial decryption key are sent to B.
- **Re-Encryption**. Algorithm $RE(rk, C)$ takes re-encryption key $rk_{A\to B}$ and ciphertext C as input, outputs ${C_1}' = {C_1}^{rk_{A\to B}} = g^{r_A s(\alpha - id_A)(\frac{\alpha - id_B}{\alpha - id_A} + a_i y)}$, and sends

the re-encrypted ciphertext C'= $(C_1{}', C_1, C_2, C_3, C_4, C_5)$ to Bob.

- **Decryption(delegatee)**. The decryption algorithm takes private key usk_B of delegatee *B*, partial decryption key and ciphertext C' as input, outputs message *m*=$C_3 \frac{e(C_1{}', {h'_B}^{(1/r_A)(1/r_B)}) {C_2}^{r_{B,1}}}{e(C_1, {B_1}^{(1/r_A)(1/r_B)})}$.

There are several security properties for proxy re-encryption scheme, we extract five important properties which are of great importance to our DRM content sharing system:

- Non-transferable: In PRE, the proxy and a set of colluding delegatees cannot re-delegate decryption rights. This is called Non-transferable. For example, from $rk_{A \to B}$, sk_B and pk_C, they cannot produce $rk_{A \to C}$.
- *Unidirectional*: Delegation from $A \to B$ does not allow delegation from $B \to A$.
- *Original-access*: *A* can decrypt re-encrypted ciphertexts that were originally sent to her.
- *Collusion-"safe"*: *B* and the proxy's collusion cannot recover *A*'s secret key.
- *Non-transitive*: Based on the re-encryption keys, $rk_{A \to B}$ and $rk_{B \to C}$, the proxy cannot produce $rk_{A \to C}$.

V. Our DRM system TC-DRM

In this section we present the Traceable and Controllable DRM Content Sharing system (TC-DRM) architecture which is shown in figure 2. The system consists of four main components: Content Provider, DRM Technology Provider, DRM Agent and Device.

Content Provider(*CP*): Content provider is an organization or individual that creates information, educational or entertainment content, and publishes them in a secure form. It coordinates the whole DRM content sharing processes. When two different parties agree to allow content sharing, *CP* is responsible for collecting information from the those parties and deriving a re-encryption key to DRM Agent.

DRM Technology Provider(*DP*): The DRM technology provider provides technologies to control use of digital media by preventing access, copying or conversion to other formats by illegal end users.

DRM Agent(*DA*): DRM agent requests a re-encryption key from *CP*, and translates encrypted contents accessible by device *a* to encrypted contents accessible by device *b*.

Device(*D*): Device has stored the public/private key of itself, as well as the public key that a device must trust: for example, the public key of *DP*.

We assume that the exporting device D_A and importing device D_B render similar content format and the exporting and importing DRM systems use similar encryption algorithms. The assumptions are reasonable, because most if not all portable music players play the MP3 formats, and most DRM systems, such as Fairplay and Windows Media DRM, use the AES encryption algorithm to encrypt their contents.

Table 1 defines the notations we use to describe the proposed system.

TABLE I
Notations

M	Plaintext of protected content
C	Ciphertext
C'	Re-encrypted Ciphertext
k	Content encryption key
usk	Private key
upk	Public key
$S(usk, m)$	Sign on message *m* with private key *usk*
σ	Signature
$E_k(\cdot)$	Symmetric encryption with key *k*
$AE(upk, m)$	Asymmetric encryption on information *m* with public key *upk*
$rk_{a \to b}$	Re-encryption key used to re-encrypt a message encrypted under *upk* of entity *a* to one under *upk* of entity *b*
$RE(rk, C)$	Re-encryption on information *C* using key *rk*

In our TC-DRM system, a DRM content shareable protocol consists of three phases: Initialization Phase, Content Usage Phase, and Content Sharing Phase.

Initialization Phase When DRM Technology Provider DP_A agrees on a contract that its protected content can be played in device b (D_b) protected by DRM Technology Provider DP_B, DP_B should make some of D_b's private information h'_b usable to DP_A. This action would not reveal D_b's private key. Further, DP_A chooses a random value $a_i \in Z_p$ and sends it to CP for generating re-encryption key in content sharing phase.

Content Usage Phase

1) Consumer purchases an encrypted digital content $E_k(M)$, $C = AE(upk_{D_a}, k)$ from *CP* and stores it in its *DA*.
2) D_a decrypts C to get *k* with its private key. Then with *k*, D_a can decrypt $E_k(M)$ to access the content *M*. Note that D_a is just able to use *k* for decrypting $E_k(M)$ in the device, but unable to leak *k* outside.

Content Sharing Phase

1) When consumer wishes to play the content in his another device D_b which is manufactured under DP_B's DRM standard, consumer sends a request along with D_b and its server DP_B's information to DA.
2) *DA* sends a content sharing request to DP_A with identity information of D_b and DP_B.
3) DP_A signs D_b's identity ID_b as $S(usk_{DP_A}, ID_b)$. Then DP_A sends the signature σ, ID_b to *CP* via a secure channel to request for a re-encryption key $rk_{a \to b}$.
4) CP extracts D_b's *ID* from signature, and verifies DP_A's signature. If verification passes, CP generates a unique randomly-selected secret parameter $y \in Z_p$, and outputs a re-encryption key $rk_{a \to b}$=$(\frac{\alpha - id_a}{\alpha - id_b} + a_i y) \bmod p$ and B_1=$\left({h_1}^{r_b} g^{-r'_b}\right)^{a_i y/(\alpha - id_b)}$. *CP* sends $rk_{a \to b}$ to DA and B_1 to DP_A.
5) *DA* re-encrypts C, and sends re-encrypted data $C' = RE(rk_{a \to b}, C)$ to D_b.
6) DP_A generates ${h'_b}^{1/r_a}$ and ${B_1}^{1/r_a}$, sends them to D_b as the license.

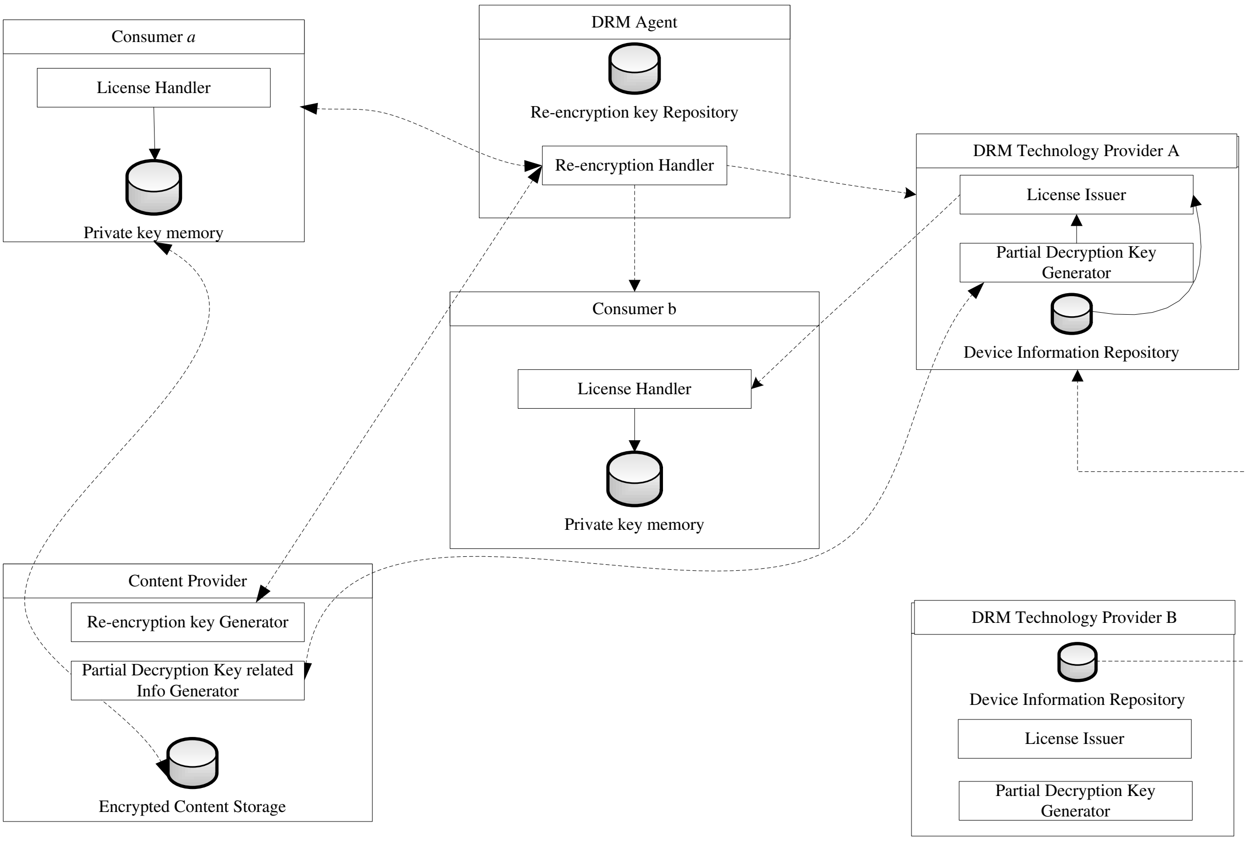
Consumer *a*
License Handler
Private key memory
DRM Agent
Re-encryption key Repository
Re-encryption Handler
DRM Technology Provider A
License Issuer
Partial Decryption Key Generator
Device Information Repository
Consumer b
License Handler
Private key memory
Content Provider
Re-encryption key Generator
Partial Decryption Key related Info Generator
Encrypted Content Storage
DRM Technology Provider B
Device Information Repository
License Issuer
Partial Decryption Key Generator

Fig. 2. System Architecture

7) D_b uses its private key and license to decrypt C' to get k. Then decrypts $E_k(M)$ to access the content.

A. Payment Scenario

The payment scenario is divided into two parts, one part is the content purchase payment, the other one is content sharing payment. The content purchase payment should be paid to *CP* by consumer when buying the content from *CP*. To encourage *CP* and *DP* to participate in the content sharing, we have to ensure that this scheme would bring benefit to them. Thus content sharing payment should be done when consumer requests content sharing service. After received the payment from consumer, *CP* gives the re-encryption key to *DA*, and DP_A sends the license to D_b.

B. Analysis

This DRM content sharing infrastructure can provide secure content sharing among different devices of the same *DP*, as well as among different devices of two different *DP*s. This system can achieve three important properties that previous re-encryption based DRM schemes cannot achieve.

- *Controllable*: The re-encryption generation must involve *CP*'s participation. Without *CP*, no one can share content with others. Further, the license is generated by DP_A. Without DP_A, D_b is unable to access the content. Thus both *CP* and DP_A control the content sharing together.
- *Non-transferable*: This property cannot be achieved in paper [12], [8], but successfully solved in our scheme. *DA* and DP_A cannot collude to generate another "re-encryption key" without asking content provider. The detailed analysis and proof can be found in [4].
- *Traceable*: Paper [8] claimed that they can track the translation of DRM content because *CP* is involved to generate a re-encryption key, so that *CP* can trace translation of its content. But in the attack described in section III of our paper, we show that malicious employees in *DP* can collude with *DIA* to generate "re-encryption" key for translating content without *CP*. Thus paper [8] cannot achieve traceable property as they claimed. In this paper, non-transferable property ensures that if without *CP*, *DA* and *DP* would be unable to generate a valid re-encryption key. Thus *CP* can always trace the content.

Due to the properties of non-transferable re-encryption scheme, the proposed DRM content sharing system can achieve four important security properties:

- *Unidirectional*: Encrypted content can be shared from D_a to D_b, does not mean that it can also be shared from D_b to D_a with the same re-encryption key.
- *Original-access*: D_a can decrypt re-encrypted content that were originally bought by it.
- *Collusion-"safe"*: *DA* and DP_A cannot collude to recover *CP*'s private key.
- *Non-transitive*: Encrypted content can be shared from D_a to D_b, and from D_b to D_c respectively, which does not mean that it can also be shared from D_a to D_c.

VI. Implementation and Evaluation

The efficiency of DRM system using non-transferable re-encryption scheme can be measured by referring to the data in [4]. Here we focus on comparing Lee's scheme [8] with ours. However, Lee [8]'s system uses the MIRACL cryptographic library [11] with 160-bit group, and our system uses PBC library [9] with 512-bit size for order of the base field. Thus various choices, such as parameter sizes and encryption granularity can greatly affect the efficiency of the scheme. To have a more accurate comparison result of scheme efficiency, we re-implement Lee's DPRE scheme using PBC library. The result shows that DPRE takes 51.27ms for encryption, and 29.72ms for re-encryption. When compared with our scheme (27.1ms for encryption and 12.6ms for re-encryption), our scheme is more efficient than DPRE. This is quite a significant reduction especially when the *DA* has to handle a large number of re-encryption requests. This shows that our system is efficient. More details can be found in [4].

VII. Conclusions and future work

In this paper we have proposed an approach to avoid illegal DRM content sharing with non-transferable re-encryption scheme. The billing method for content sharing improves the preservation of content provider's benefit. The whole system is efficient and practical.

We plan to extend this work by including a mobile phone as a semi-trusted third party. As a private belonging, mobile phone can be relied upon to do certain things for DRM purpose, such as acting as *DA* for re-encryption, and be responsible for managing payment to *CP* for each content sharing event.

Acknowledgment

The work described in this paper was partially supported by the General Research Fund from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. RGC GRF HKU 713009E), the NSFC/RGC Joint Research Scheme (Project No. N_HKU 722/09), and HKU Seed Fundings for Basic Research 200811159155 and 200911159149.

References

[1] http://en.wikipedia.org/wiki/Digital_rights_management.

[2] N. Dufft, A. Stiehler, D. Vogeley, and T. Wichmann. Digital music usage and drm. http://www.indicare.org/tiki-download_file.php?fileId=110, May 2005.

[3] A. Durand, M. Éluard, S. Lelievre, and C. Vincent. Smartpro: A smart card based digital content protection for professional workflow. In *Smart Card Research and Advanced Applications, 8th IFIP WG 8.8/11.2 International Conference, CARDIS 2008, London, UK, September 8-11, 2008. Proceedings*, Lecture Notes in Computer Science, pages 255–266. Springer, 2008.

[4] Y.-J. He, T. W. Chim, L. C. K. Hui, and S. M. Yiu. Non-transferable proxy re-encryption scheme for data dissemination control. http://eprint.iacr.org/2010/192.pdf.

[5] Y.-J. He, T. W. Chim, L. C. K. Hui, and S. M. Yiu. Non-transferable proxy re-encryption scheme for data dissemination control. *submitted.*

[6] J. Johansen. Buy drm-free songs from the itunes music store. http://www.engadget.com/2005/03/18/buy-drm-free-songs-from-the-itunes-music-store/, 2005.

[7] D. W. Kravitz and T. S. Messerges. Achieving media portability through local content translation and end-to-end rights management. In *Proceedings of the Fifth ACM Workshop on Digital Rights Management, Alexandria, VA, USA, November 7, 2005*, pages 27–36. ACM, 2005.

[8] S. Lee, H. Park, and J. Kim. A secure and mutual-profitable drm interoperability scheme. In *Proceedings of the The IEEE symposium on Computers and Communications*, ISCC '10, pages 75–80, Washington, DC, USA, 2010. IEEE Computer Society.

[9] B. Lynn. Pbc: The pairing-based cryptography library. http://crypto.stanford.edu/pbc/.

[10] D.-W. Nam, J.-S. Lee, J.-H. Kim, and K.-S. Yoon. Interlock system for drm interoperability of streaming contents. In *Consumer Electronics, 2007. ISCE 2007. IEEE International Symposium on*, pages 1–4, June 2007.

[11] M. Scott. Miracl. shamus software. http://www.shamus.ie/.

[12] G. Taban, A. A. Cárdenas, and V. D. Gligor. Towards a secure and interoperable drm architecture. In *Proceedings of the Sixth ACM Workshop on Digital Rights Management, Alexandria, VA, USA, October 30, 2006*, pages 69–78. ACM, 2006.