HAL
archives-ouvertes.fr

Efficient and Enhanced Solutions for Content Sharing in DRM Systems

Michal Davidson, Ehud Gudes, Tamir Tassa

▶ To cite this version:

Michal Davidson, Ehud Gudes, Tamir Tassa. Efficient and Enhanced Solutions for Content Sharing in DRM Systems. 28th IFIP Annual Conference on Data and Applications Security and Privacy (DBSec), Jul 2014, Vienna, Austria. pp.373-381, 10.1007/978-3-662-43936-4_25 . hal-01284873

HAL Id: hal-01284873
https://hal.inria.fr/hal-01284873

Submitted on 8 Mar 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution| 4.0 International License

Efficient and Enhanced Solutions for Content Sharing in DRM Systems

Michal Davidson[1], Ehud Gudes[2] and Tamir Tassa[1]

[1] The Open University, Ra'anana, 43100, Israel, michalsaraw@gmail.com, tamirta@openu.ac.il
[2] Ben-Gurion University, Beer-Sheva, 84105, Israel, ehud@cs.bgu.ac.il

Abstract. We present a solution to the problem of content sharing in digital rights management (DRM) systems. Users in DRM systems purchase content from content providers and then wish to distribute it between their own devices or to other users. The goal is to allow the sharing of such content, with the control of the content provider, while ensuring that it complies with the content's usage rules. While most of the previous studies on content sharing in DRM systems assume the existence of authorized domains, ours does not make that assumption. The solutions that we present here are based on Certified Sharing Requests which are used when devices request from the content provider authorization to share content with other devices. Our solutions enhance the usability of DRM, from both the users' and content provider's perspective, by supporting on-the-fly sharing, sharing and re-sharing of controlled content, and a pay-per-share business model.

Keywords: digital rights management, content sharing, authorized domain, proxy re-encryption

1 Introduction

The usage of Digital Rights Management (DRM) in the digital media industry is controversial, since it limits the use of legally purchased content, and it does not allow certain scenarios that were previously possible. One of the main controversies with DRM systems is with regard to content sharing. When physical or DRM-free content is purchased it can be shared, copied, and re-sold. On the other hand, in DRM systems, the content provider (CP) wants to control such content sharing, and ideally would like to get paid whenever such content is further shared with other users.

Most current solutions for content sharing propose and expand upon the use of an "Authorized Domain" — a group of devices which can freely share content between themselves [1]. However, such solutions have two main drawbacks: they do not support "on the fly" sharing, namely, sharing between devices that do not belong to the same domain; and they do not offer means to control which content can be shared between two devices.

In this paper we propose solutions for content sharing that do not rely upon authorized domains. A recent scheme that solves the content sharing problem without assuming authorized domains was proposed by Ma et al. [2]. Their scheme uses a proxy re-encryption method [3] which allows re-encryption of a message without decrypting it first. Although this method is elegant and secure (see its detailed description in the next section), it involves a considerable overhead in terms of storage, and it relies on the complex cryptographic primitive of bilinear pairing. Moreover, the implementation of the pairing in [2] dictates using the El-Gamal public key cryptosystem and prevents using other public key methods like the prevalent RSA cryptosystem. Finally, the solution in [2] does not support re-sharing of purchased content with other users, or a flexible payment scheme.

Here we address the above two problems and present a simpler scheme for controlled sharing in DRM systems. Our scheme is called the *Certified Sharing Request (CSR) Scheme*. It supports "on the fly" sharing, re-sharing to any preset depth, verification of content-dependent sharing privileges, CP knowledge of sharing, and a pay-per-share business model. We achieve those functional objectives while ensuring common security and privacy properties.

2 Background and related work

2.1 Definitions

Digital Rights Management ("DRM") is a method for controlling the viewing and distribution of digital content. A DRM system consists of the following entities:

- Content (C) - a purchasable item of digital content. The content is distributed in an encrypted format, using a symmetric encryption, and can only be decrypted using the corresponding content key.
- Content License (CL)- a record that includes the content key and a set of usage rules. Content licenses are typically encrypted using public key encryption.
- Content Provider (CP) - The entity that owns the content items and wishes to control the distribution of the content to its client devices.
- Device (will be denoted by A, B, A_0, A_1 etc.) - a tamper proof computer processing unit that is capable of parsing and decrypting the encrypted content and the encrypted content licenses. Each device holds a secret key and a corresponding certificate, which is signed by a Certificate Authority. We assume that the device's secret key, as well as the content keys which the device extracts from content licenses, are securely stored and processed in a trusted hardware device (so called Trusted Computing Base, or TCB) and cannot be accessed by a third party.

2.2 Related work and content sharing with proxy re-encryption

Most literature on the topic of content sharing within DRM systems focuses on the use of the Authorized Domain model [1]. This is the classic DRM solution for content sharing, in which a group of authorized devices are defined as belonging to a joint domain, and devices within the same domain can freely share content between them.

The studies [4, 5] suggest improvements in the authorized domain model. Other studies do not assume that model: Sadeghi et al. [6] provide a secure platform on open systems which allows the usage of dynamic licenses; Lee et al. [7] propose a system for content sharing which relies on time-based rights.

A recent work on DRM and content sharing [2] uses the method of proxy re-encryption [3]. The method in [3] allows users who received a message that was encrypted with their public key to re-encrypt it for other users without decrypting it first. They describe two types of probabilistic public key encryption functions — *first* and *second level encryptions*. If (sk_A, pk_A) denotes the private and public key pair of user A, then $E_\ell(m, pk_A)$, $\ell = 1, 2$, denote the first and second level encryptions of the plaintext m for user A. User A may decrypt any ciphertext in $E_\ell(m, pk_A)$ using his private key sk_A. In addition, he may re-encrypt $E_2(m, pk_A)$ into $E_1(m, pk_B)$ without decrypting it first. Their method uses bilinear pairings that are based on the Tate pairing [8]. We now proceed to describe the content sharing solution of [2].

Purchasing content. When device A requests from the CP to purchase content C, the CP sends to A a message $x \in E_1(m, pk_A)$, where $m = CL$ is the content license of the requested content. In addition, the CP generates a random key pair (sk_R, pk_R) and sends to A a message $y \in E_2(m, pk_R)$. Finally, it adds to its records a new record that holds the identifiers of A and C, the generated random key pair, and a counter of the number of times in which A shared C so far. After the purchasing protocol is completed, A uses the message x to recover the content license CL, with which it can decrypt the encrypted content.

Sharing content. When device A wishes to share the purchased content C with another device B, it sends a corresponding request to the CP. The CP checks the details of the two devices A and B, and the number of times in which A had already shared that particular content. If that sharing request is approved, the CP computes a re-encryption key, rk, using B's public key pk_B and the random private key sk_R that was generated when A purchased that content, and sends it to A. A uses rk together with the message y which it received upon purchasing that content in order to compute a ciphertext $z \in E_2(m, pk_B)$, by means of bilinear pairing. Device A then sends z together with the encrypted content to device B. B proceeds to recover the content license m and decrypt the content.

There are several disadvantages to this solution: (a) Payment for sharing content can only be performed by the device A who is sharing the content; a better business model would be for the device B to pay to the CP for the shared content. (b) This model requires that the CP stores a record for each device and each purchased content, where each record stores the corresponding counter and a pair of cryptographic keys. (c) The method is limited to only one

level of sharing; it does not allow device B to re-share the content with another device. (d) The method relies upon the complex and costly bilinear pairing function. (e) The usage of bilinear pairings in [2] is based on El-Gamal public key cryptography and, thus, prevents using other public key cryptosystems, such as RSA.

3 Certified Sharing Requests (CSR) and the CSR scheme

Here we present our solution for content purchasing, sharing and re-sharing. We describe how a given device A_0 can purchase content C from the CP; how A_0 may share C with another device A_1; how A_1 can re-share C with A_2; and, in general, how to perform re-sharing of any depth.

While in the proxy re-encryption solution A_0 re-encrypts the content licence for A_1, in our solution the CP encrypts the content license for A_1. We chose to transfer the task of encrypting the content license from A_0 to the CP for the following reasons: (a) The CP must be involved in any such sharing or re-sharing operation since it needs to verify that the sharing or re-sharing is consistent with the usage rules for the content C. Hence the CP can also encrypt the content license for the new device. (b) In [2], the process of re-encryption can be performed only once per content and device, and thus does not support re-sharing. In our scheme, the CP can perform a direct encryption rather than re-encryption, hence re-sharing of any depth is possible.

Our solution is based on Certified Sharing Requests (CSRs). The CSRs include: information on the content C, the certificates of the devices that are involved in the sharing and re-sharing operations, and payment information. The CSRs are signed by all involved devices. The mechanism of CSRs is flexible enough to support interoperability between DRM systems; namely, a device in one DRM system can share content with a device that belongs to a different DRM system, under the above assumptions. This will allow content providers to charge devices for "pay per sharing", regardless of the DRM system to which they belong.

3.1 Purchasing content

Here we describe the process that takes place when a device, A_0, wishes to purchase a certain content, C. At the completion of this process, A_0 receives from the CP three items: (a) the content C, encrypted by a symmetric encryption using the content key k_C; (b) the content license (which includes k_C), encrypted by A_0's public key; and (c) a corresponding sharing license, denoted SL, which will be used only when A_0 chooses to share C with other devices. Note that in the entire paper the content license and the content itself can be decrypted only by the TCB, so no key in the clear (i.e., non-encrypted key) can be sent out by the device.

When A_0 wishes to purchase a content C, the following protocol is executed:

1. A_0 sends to the CP a message with A_0's certificate and C's ID.

2. The CP verifies A_0's certificate, and that it is not revoked, then encrypts the content license $m =$ CL of C with A_0's public key, creates a corresponding sharing license SL, and certifies it by signing it. The signed sharing license will be denoted by $[SL, Sig_{CP}]$. The CP then sends to A_0 the encrypted content, the encrypted content license, and the signed sharing license.
3. The CP creates and stores a record of the form (A_0, C, ST_{C,A_0}), where ST_{C,A_0} is a counter of the number of times in which A_0 shared the content C with other devices; it is initialized to zero.
4. A_0 also creates a counter ST_C for the number of times that it shared the content C with other devices.

The sharing license SL which the purchasing device A_0 receives upon purchasing the content C will be used when A_0 wishes to share that content with another device A_1. The SL will contain the following fields: (1) C (the ID of the purchased content); (2) GDI (the Global Device ID of A_0); (3) MSL (Maximum Sharing Level), denoting the depth of permitted sharing for C; and (4) NoS (Number of Sharings), bounding the total number of devices that can receive the shared content from A_0 by means of sharing or re-sharing.

3.2 Sharing content

When A_0 wishes to share C with A_1, the following protocol is executed.

1. A_0 sends to A_1 the message $[SL, Sig_{CP}, COD]$ where $[SL, Sig_{CP}]$ is the certified sharing license that A_0 received from the CP when it purchased C, and COD (Charge Original Device) is a field that indicates which device will be charged for the content sharing: $COD = 0$ means that A_0 will be charged, while $COD = 1$ means that A_1 will be charged.
2. A_1 adds to the received message its certificate $Cert_{A_1}$ and a payment information field EPI. EPI includes the information needed to charge A_1 for performing this sharing, if $COD = 1$; otherwise, if $COD = 0$, EPI is empty.
3. A_1 sends to A_0 the message $[SL, Sig_{CP}, COD, Cert_{A_1}, EPI, Sig_{A_1}]$, where the last field is A_1's signature on the preceding fields in the message.
4. A_0 verifies that the internal counter ST_C which it maintains for the content C is smaller than NoS (the value that appears in SL), and that A_1 did not alter the value COD. If those verifications passed successfully then A_0 increments ST_C and signs the sharing request. The result is called a CSR (Certified Sharing Request):

$$CSR_1 := [SL, Sig_{CP}, COD, Cert_{A_1}, EPI, Sig_{A_1}, Sig_{A_0}] .$$

   Here, Sig_{A_0} is the signature of A_0 on all preceding fields in CSR_1.
5. A_0 sends CSR_1 to the CP.
6. The CP performs the following verifications: (a) the three signatures in CSR_1; (b) neither of the devices A_0, A_1 is revoked; (c) the MSL field, as appears in SL, is at least 1; and (d) it retrieves the record (A_0, C, ST_{C,A_0}) and checks that $ST_{C,A_0} < NoS$ (and if so, it increments the value of ST_{C,A_0}). If all verifications were successful, the CP encrypts the content license with A_1's public key and sends it to A_1 (either via A_0 or directly).

7. A_0 sends to A_1 the encrypted content. (Recall that A_0 already has the encrypted content, since it received it from the CP upon purchasing it.)
8. A_1 decrypts the content license using his private key in order to recover the content key. It then proceeds to decrypt the content using that key.

Note that the device A_0 has to verify the internal counter ST_C in Step 4 in the protocol above in order to refrain from unnecessary communications vis-a-vis the CP. The CP repeats the same check (Step 6d) since it does not trust A_0, but, as explained above, the check by A_0 in Step 4 is still necessary for communication overhead considerations. We note that if the check in the device is performed in the TCB, it can prevent Denial of Service attacks.

3.3 Re-sharing content

Assume that a specific content C was already shared by the following chain of devices, $A_0, A_1, \ldots, A_{i-1}$; i.e., A_0 purchased that content from the CP, and then it shared it with A_1, who continued to share it with A_2, and so forth. Below we describe how A_{i-1} can re-share the content with a new device A_i. Before doing so, we comment that if the field COD equals 0, then, as before, it means that A_0 will pay for that re-share; however, if $COD = 1$ then the recipient of the content in the re-sharing operation (i.e., A_i in this case) will be charged for that operation.

The re-sharing protocol proceeds as follows:

1. Device A_{i-1}, that already possesses $[SL, Sig_{CP}, COD, Cert_{A_1}, \ldots, Cert_{A_{i-1}}]$ from the protocol that took place when it obtained the shared content, sends this sharing request to A_i.
2. A_i adds to the sharing request its certificate $Cert_{A_i}$ and the payment information field EPI (where $COD = 1$) or an empty EPI (where $COD = 0$).
3. A_i sends to A_{i-1} the sharing request

$$[SL, Sig_{CP}, COD, Cert_{A_1}, \ldots, Cert_{A_{i-1}}, Cert_{A_i}, EPI, Sig_{A_i}]\,,$$

   where the last field is A_i's signature on the preceding fields in the message.
4. The message is sent up the chain of devices, where device A_j adds its own signature on the message that it received from A_{j+1}, and then sends it to A_{j-1}, $j = i-1, \ldots, 1$.
5. A_0 verifies that ST_C is smaller than NoS, and that A_i did not alter the value COD. If those verifications passed successfully then A_0 increments ST_C and signs the sharing request. The resulting CSR is:

$$CSR_i := [SL, Sig_{CP}, COD, Cert_{A_1}, \ldots, Cert_{A_i}, EPI, Sig_{A_i}, \ldots, Sig_{A_0}]\,.$$

6. A_0 sends CSR_i to the CP.
7. The CP performs the following verifications: (a) it verifies all $i+2$ signatures that appear in CSR_i; (b) none of the devices $A_0, \ldots, A_i$ is revoked; (c) the MSL field, as appears in SL, is at least i; and (d) it retrieves the record (A_0, C, ST_{C,A_0}) and checks that $ST_{C,A_0} < NoS$ (and if so, it increments the value of ST_{C,A_0}).

8. If all verifications were successful, the CP encrypts the content license with A_i's public key and sends it to A_i (either via A_0 or directly).
9. A_{i-1} sends to A_i the encrypted content. (Recall that A_{i-1} already has the encrypted content, since A_{i-2} shared it with him.)
10. A_i decrypts the content license using his private key in order to recover the content key. It then proceeds to decrypt the content using that key.

3.4 The partial trust scenario

Due to space limitations, we focused in this paper on a scenario of non-trust, where devices that are direct clients of the CP are not granted any trust regarding the right to authorize a sharing request. The CP does not delegate to A_0 any verification tasks; it performs all necessary checks before sending out the content license encrypted for the new device. The disadvantage in such a non-trust scenario is greater storage and performance overhead. In the full version of this paper we present a relaxed scenario of partial trust, in which the CP has a partial trust in its direct client devices. In particular, device A_0 is trusted to do most of the verification which is currently done by the CP in the full-trust scenario. Therefore, in such a model, the storage and computational costs for the CP are reduced significantly.

3.5 Advantages and disadvantages of the CSR scheme

The CSR scheme supports re-sharing, where the depth and number of re-sharings can be set upfront and controlled. Payment can be made by either the device which originally purchased the content from the CP or from the device which is the recipient in the re-sharing act, what allows a more flexible pay-per-share business model. Compared to the proxy re-encryption scheme, the CP has to store a smaller database that holds only the counter per device per content, without the need to store a pair of cryptographic keys. In the partial trust scenario the amount of data that needs to be stored by the CP is further reduced, since the CP has to store just one counter per device (and not per device per content). Finally, the CSR scheme does not rely upon the complex and costly bilinear pairing function.

When comparing the security of the proxy-reencryption and the CSR scheme, we see that in both schemes public key encryption is used to protect the content license which contains the content key. In addition, the sharing requests in the CSR scheme are always signed, by the tamper-proof TCB, so that they are authenticated and cannot be repudiated. This is an advantage over the proxy re-encryption solution which does not use signatures, and thus is vulnerable to both man-in-the-middle attacks, and Denial of Service (DOS) attacks. Both solutions require each device to have a trusted computing base (TCB). The TCB of A_0 is not required by the proxy re-encryption solution for content sharing, since the re-encryption can be performed on the device itself without exposing either the content key or the device's secret key. In the CSR scheme, on the other hand, any action that involves signatures is a TCB operation. Hence, both

schemes depend on a TCB, but in the CSR scheme the overhead on the TCB is greater due to the use of signatures. The conclusion is that the security of both schemes is comparable, but the CSR scheme is advantageous in terms of protecting sensitive information on the CP, in providing authentication and non-repudiation to sharing requests, and in reducing the risk of DOS and man-in-the middle attacks.

4 Conclusions and future work

We proposed a scheme for content sharing in a DRM system. Content sharing is performed using a Certified Sharing Request (CSR). In contrast to most related work on content sharing in DRM systems, our approach does not rely on the Authorized Domain model. The CSR model improves upon the limitations of the Authorized Domain model by supporting "on-the-fly" sharing, controlled content sharing and re-sharing, and a pay-per-share business model. We propose versions of our CSR scheme both for the fully secure non-trust scenario and for the partial trust scenario with improved performance.

In the future, we would like to enhance the CSR scheme by supporting privacy preservation, i.e. allowing content sharing where the content provider receives payment and sends a content license while remaining oblivious of the identity of the user who purchased that content. We intend to investigate the application of proxy re-encryption in order to support privacy preservation within the CSR scheme. We may also refine the payment model, and define in more depth the rules whereby the content provider should halt sharing in the partial trust scenario.

References

1. Popescu, B.C., Crispo, B., Tanenbaum, A.S., Kamperman, F.: A drm security architecture for home networks. In: Digital Rights Management Workshop. (2004) 1–10
2. Ma, G., Pei, Q., Jiang, X., Wang, Y.: A proxy re-encryption based sharing model for drm. Int'l J. of Digital Content Tech. and its Applications **5**(11) (2011) 385
3. Ateniese, G., Fu, K., Green, M., Hohenberger, S.: Improved proxy re-encryption schemes with applications to secure distributed storage. ACM Trans. Inf. Syst. Secur. **9**(1) (2006) 1–30
4. Abbadi, I.M.: Digital rights management using a master control device. In: ASIAN. (2007) 126–141
5. Sheppard, N.P., Safavi-Naini, R.: Sharing digital rights with domain licensing. In: The ACM Workshop on Multimedia Content Protection and Security. (2006) 3–12
6. Sadeghi, A.R., Wolf, M., Stüble, C., Asokan, N., Ekberg, J.E.: Enabling fairer digital rights management with trusted computing. In: ISC. (2007) 53–70
7. Lee, S., Kim, J., Hong, S.J.: Redistributing time-based rights between consumer devices for content sharing in drm system. Int. J. Inf. Sec. **8**(4) (2009) 263–273
8. Galbraith, S.D., Harrison, K., Soldera, D.: Implementing the tate pairing. In: ANTS. (2002) 324–337