Scalable and Secure Architecture for Digital Content Distribution

Valer Bocan

Department of Computer Science and Engineering
“Politehnica” University of Timisoara
2 V. Parvan Ave., 300223 Timisoara, ROMANIA
vbocan@dataman.ro

Mihai Fagadar-Cosma

Information Technology Department
Alcatel Romania
9 Gh. Lazar Ave., 300081 Timisoara, ROMANIA
mihai.fagadar@alcatel.ro

***Abstract*: This paper describes a scalable and secure architecture for digital content distribution. Our architecture enables secure cooperation between content providers which share distribution rights, allowing a greater flexibility in real-world scenarios. Furthermore, the new architecture is highly scalable as it enables content providers to simultaneously service several clients who request the same content.**

***Index Terms* – scalable DRM, digital content, distribution, redistribution, secure broadcast, cryptography, secret splitting.**

I. Introduction

Digital content distribution has become a widely discussed topic in the past years, due to the increasing popularity of the Internet and of the personal devices capable of playing digital multimedia content. More and more providers offer their customers the possibility to access, for a fee, large on-line databases of multimedia content which they can download on their personal devices.

Due to integration of technologies like Bluetooth or Wi-Fi, personal devices have become capable of sharing information among them in a point-to-point manner. This has opened the way to digital content redistribution, a process which generates revenue loss by excluding the content provider from the data transfer.

To preserve the digital content from illegal copying and unauthorized distribution and to ensure that copyright laws are respected, content providers have searched for new ways of implementing secure distribution systems, based on Digital Rights Management (DRM) policies. The greatest challenge posed to such a system is to ensure that these policies are effective even after the customer came into the possession of the digital content, especially when he attempts to redistribute it to another user.

The current approach to the above-mentioned problem has been to introduce the notion of *compliant devices*, which, by their design, guarantee to respect the DRM policies associated with the multimedia content they are playing. For example, a compliant device will refuse to share its contents with a *non-compliant device*, or to redistribute the multimedia content to another compliant device if the associated DRM policy forbids it to do so.

Systems which allow digital content redistribution by enforcing the DRM policies at client level on peer-to-peer networks have already been proposed [1]. However, they are not scalable, and present a vulnerability to DoS attacks, which may render them inoperable. These systems also rely on the fact that each content provider has all the necessary rights to distribute the multimedia content to its customers, which may not always be true in real situations. Such rights may be distributed among several content providers.

In the present paper we propose a scalable digital content distribution system, which relies on secure broadcast for distributing the digital content to several clients simultaneously. Thus the sever load is decreased considerably, while maintaining the same degree of security as in normal point-to-point connections.

We also propose a mechanism to remove the limitation of a single content provider which has all the rights over the digital content. In our architecture, for a digital content, there may be several content providers which share the distribution rights over that content, and a common consent is required to distribute it. The paper is organized in six sections, as follows. In Section II we present the system architecture and the parties involved in the digital content distribution and redistribution. Section III focuses on the cryptographic techniques used to secure the communication between the system components. Section IV describes the scalable distribution of digital content in both cases: provider to client and client to client, and how the techniques presented in Section III apply to our particular case. Section V analyzes the possible threats to this system and finally, Section VI presents the advantages of the proposed solution and draws the final conclusions.

II. System Architecture

The system architecture, presented in Fig. 1, contains two major parts:

- The authority and content distribution part composed of one *Master Content Provider* and several authorized *Content Providers*;
- The consumer network composed of *Clients* or *Content Proxies*.

The parties involved in the digital content distribution scheme are as follows:

A. Content Providers

Content providers (CPs) are parties which share the rights to distribute the digital content to the consumer network. A consumer which desires to receive the digital content will issue a request to a content provider which, in turn will subject the request to the approval of all the CPs which share the distribution rights of the digital content. If all CPs approve the request, then the content provider will send the digital content to the client, otherwise it will reject the request.

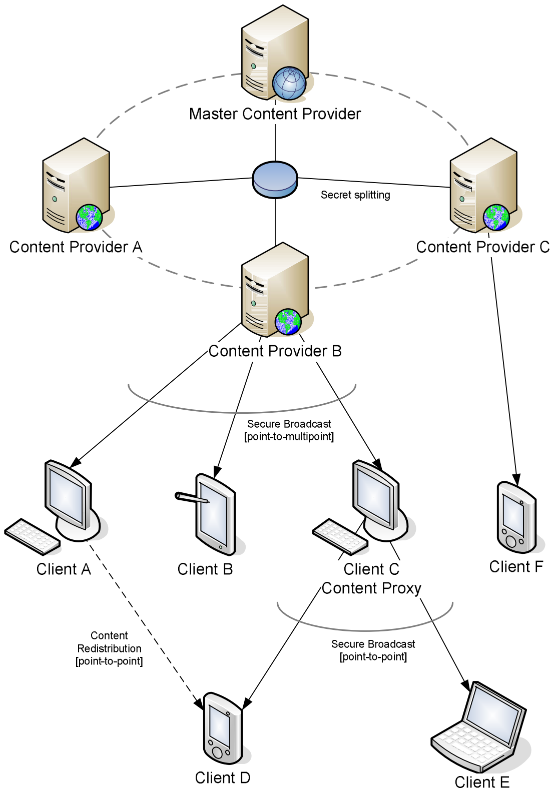

Fig. 1. The content distribution system architecture

A single piece of digital content will be broadcast to the clients who request it in a given time frame. This technique greatly reduces the load on the server by servicing several clients at a time.

B. Master Content Provider

The Master Content Provider (MCP) represents the organization which controls the activity of all content providers (CP), and it is involved indirectly in the content distribution process. It can be regarded as the authority which supervises the request approval process between the content providers and generates session keys for their activities.

C. Clients

Clients are compliant devices which have the right to play the multimedia content received from a content provider, and may optionally purchase the rights to redistribute it. The requirements which must be fulfilled by such a compliant device are outlined in the Trusted Platform Module (TPM) specifications [4]. Each compliant device is endowed at manufacturing time with a pair of keys: a *public key* and a *private key*, which it uses when exchanging information with a CP or another client.

The redistribution process, illustrated in Fig. 1 between clients A and D, takes place in accordance to the DRM policies associated with the content (e.g. the content may be redistributed only a limited number of times and to a limited number of compliant devices). If DRM policies are not respected by the client, the CP may revoke its redistribution rights, by using the device revocation mechanism presented in [1].

D. Content Proxies

Content proxies differentiate from clients in the way that they are acting as relays between the CP and other customers. Any client may become a content proxy if he desires to do so, by signaling this intent to the CP. Since the multimedia content is encrypted, the proxy will not be able to render it if not addressed to it directly, but is able to forward the content to its legitimate destination. In Fig. 1, Client C has the role of content proxy and it distributes the data to clients D and E, through secure broadcast.

By making use of content proxies, a CP can reduce its server load significantly, since client requests for digital content can be serviced by the proxy, without the involvement of content providers. The CP constantly monitors proxy activities to make sure that they are in accordance with the DRM policies.

III. Cryptographic Techniques

Data transfers between the parties involved in the proposed content distribution scheme take place on secure channels, protected by a series of cryptographic methods, as follows: the cooperation between the CPs is implemented by using the secret splitting technique [2] while the CP to client and proxy-to-client data transfers are based on secure broadcast with secure locks [3] generated using the Chinese Remainder Theorem [5].

A. Secret Splitting Technique

As we previously stated, the content providers may share distribution rights, in which case they must all approve the client requests. Therefore, the CPs must cooperate for each request approval, in a secure manner.

The most secure way to make the CPs cooperate is to share a secret among them, by using the secret splitting technique [2]. This way, no content provider can obtain the secret without the help of the other CPs with whom it shares that secret.

Considering a secret message *M* of length *m* and a group of *n* secret sharers, designated P_1, P_2, ..., P_n, the secret can be split among the *n* sharers as follows:

1) First, a number of *n-1* random bit strings of length *m*, R_1, R_2, ... , R_{n-1} are generated.
2) The message *M* is encrypted, resulting the secret $S = M \otimes R_1 \otimes R_2 \otimes K \otimes R_{n-1}$.
3) The secret *S* is distributed to P_1, R_1 is distributed to P_2, R_2 to P_3 and so on, up to R_{n-1} which is distributed to P_n.

It is obvious that the only way to obtain the secret *M* is by XOR-ing together the pieces distributed among the secret sharers. The sharers themselves need not even know who received *S*, and who received the random strings R_i. This makes the secret splitting technique absolutely secure.

In our case, considering that the distribution scheme contains one *MCP* and a number of *n* CPs: CP_1, CP_2, ..., CP_n, which share the distribution rights for the digital content, the secret splitting technique works as follows:

1) CP_i receives a client request.
2) If CP_i accepts the request, it asks the *MCP*, which is the trusted authority, to create and distribute a secret message *M* among all the CPs.
3) The *MCP* generates the secret message *M,* known only to himself, splits it in *n* pieces, and shares it among all CPs, by using the secret sharing technique.
4) CP_i forwards the client request to all the other CPs.
5) If a CP agrees to the client request, it will share its part of the secret to CP_i, otherwise it will send an empty string of bytes instead.
6) Based on the answers from the rest of the secret sharers, CP_i will attempt to reconstruct the message *M*, which will be sent to the *MCP* for validation. Only if all the other CPs agreed with the requests, CP_i will be able to reconstruct the message *M.*
7) The *MCP* will compare the message decrypted by CP_i to the original message *M*, and if they match, it will authorize CP_i to distribute the digital content to the client, otherwise it will instruct it to reject the request.

B. Using the Chinese Reminder Theorem to generate secure locks

A previous research on digital content distribution [1] required that for each request the CP encrypts the content using the public key of the client which made the request. A number of *n* clients requesting content from the CP will require *n* content encryptions, even if the content may be the same for all clients. This places a serious burden on the system that services incoming requests and therefore represents a scalability limitation.

In order to reduce the server load, we propose that the CP use secure broadcasting techniques to send the data to all clients which requested it. In this case, the digital content is encrypted only once, using an encrypting session key known only to the CP, and sent along with a decrypting session key to the clients. To make sure that only legitimate clients can obtain the decrypting key, a secure lock is needed to protect it. The lock can be removed only by the legitimate clients and is generated by applying a technique known as the Chinese Remainder Theorem (CRT) [3]:

Let *C* be a group of *n* clients C_1, C_2, ..., C_n serviced by a CP, each having a pair of keys e_i (secret) and d_i (public), and N_1, N_2, ..., N_n a set of *n* positive integers that are mutually prime and publicly known in the system. From the group *C*, a subset of *k* clients ($k \geq 2$) request the same digital content from the CP. If we furthermore consider a set of *k* positive integers R_1, R_2, ..., R_k , the CRT theorem states that the system of congruencies:

$$\begin{cases} X \equiv R_1 (\operatorname{mod} N_1) \\ X \equiv R_2 (\operatorname{mod} N_2) \\ \Lambda \\ X \equiv R_k (\operatorname{mod} N_k) \end{cases} \tag{1}$$

has a common solution *X*, given by equation (2):

$$X = (\sum_{i=1}^{k} \frac{L}{N_i} \cdot R_i \cdot f_i) \operatorname{mod} L \tag{2}$$

where:

$$1 \equiv f_i \cdot \frac{L}{N_i} \operatorname{mod} N_i \tag{3}$$

with *L* being defined as $\prod_{i=i}^{k} N_i$.

For our proposed scheme, the CP will generate a pair of session keys, e_s and d_S, and encrypt the digital content only once, using e_S. The CP will also generate the R_1, R_2, ... , R_k numbers by encoding the decrypting session key d_S with the public encrypting key e_i of each requester, as follows: $R_i = Enc_{e_i}(d_S)$.

The secure lock *X* is obtained by solving the equation system (1), and sent to the clients along with the encrypted digital content. Each client *i* from the group which requested the digital content can obtain R_i from the received lock *X*, according to equation system (1), and by decrypting it with its private key d_i, can obtain d_S. Once a client has d_S, it can decrypt the multimedia content and use it according to its associated DRM policy.

From the technique described above, it can be seen that the lock cannot be broken by illegitimate clients. Even if such a client obtains a remainder R_i, it cannot extract d_S, as it does not posses the private key d_i of the legitimate client *i*. Therefore, this method is secure.

IV. Scalable Distribution of Digital Content

The content distribution process can be split into two parts:

- The content provider CP distributes the content and its associated licenses to clients (C_1)
- The client (C_1) redistributes the content to another client (C_2).

Let's introduce some notations:
e_A/d_A – the public/private key pair of entity A
$[D]e_A$ – data D encrypted under public key of A
$[D]d_A$ – data D signed with the private key of A
$[D]_K$ – data D encrypted using a symmetric key K
h(D) – a collision-free hash function h applied on data D

A. Content Provider (CP) distributes content to clients

In this part, the content provider (CP_1) distributes the digital content M and associated rights R to the clients (C_i) with permission from other content providers (CP_2, …, CP_n) and under the supervision of the master content provider (MCP).

(1) $C_1, C_2, \dots, C_n \rightarrow CP_1$: request content
(2) $C_1, C_2, \dots, C_n \leftrightarrow CP_1$: mutual authentication
(3) $CP_1 \leftrightarrow CP_2, \dots, CP_n$, MCP: agreement upon distribution rights
(4) $C_1, C_2, \dots, C_n \leftrightarrow CP_1$: payment (optional)
(5) $CP_1 \rightarrow C_1, C_2, \dots, C_n : [M]_K, [K]_{eS}, X, \eta, \delta, \Lambda$

The content provider waits for incoming requests and services them at certain intervals. Similar requests are grouped together in step (1) and serviced in the same time in order the solution to scale by decreasing the server load. In step (2), the content provider and the clients authenticate each other. Unlike the architecture described in [1], we do not perform payment at this stage as the content provider that received the requests is not yet authorized by the peers to distribute the content. In step (3), the content providers agree upon authorizing the content provider who received the requests and in step (4) the payment is performed.

In step (5), the content provider generates the secret lock X, encrypts the content M with a one-time symmetric key K, encrypts K with the session key e_S and then sends the encrypted content along with the lock X, rights η, metadata δ and the content license Λ. The rights η is a quantity that describes how the content is to be handled by compliant devices and the metadata δ associated with the content (name of the artist, the album, the song title, bit rate, etc.). The content license Λ is defined as:

$$\Lambda = [h(M, \eta, \delta, X)]_{dCP}$$

The purpose of Λ is to certify that the client has been granted rights η with respect to content M.

Rights η can be represented using authorization and access policy languages such as XACML [6] and XrML [7].

The great advantage of our architecture is that it is highly scalable. When certain content is highly demanded, several different clients may request it almost simultaneously. The content provider waits for a short time so that several requests accumulate, and then with a single encryption operation it services all clients.

B. Clients redistribute content

The rights η originally granted by the content provider may allow the client C_1 to redistribute the content to another client C_2, following the protocol below (similar to [1]):

(1) $C_2 \rightarrow C_1$: request content
(2) $C_2 \leftrightarrow C_1$: mutual authentication
(3) $C_1 \rightarrow C_2$: $[M]_{K'}, [K']_{eC2}, \eta, \eta', \delta, \Lambda, \Lambda'$
(4) $C_2 \leftrightarrow C_1$: check δ, payment (optional)
(5) $C_2 \rightarrow C_1$: φ

C_2 starts the transaction in step (1) by requesting a particular content item. In step (2), the two parties authenticate each other using their public/private key pairs. If the authentication is successful, C_1 decrypts the content of the requested item, generates a temporary symmetric key K' and encrypts the content with this key. The K' key is then encrypted with C_2's public key. C_1 then sends the new encrypted content, the session key K encrypted with C_2's public key, the original rights η and the new rights η' granted to C_2. Also, C_1 sends the original license Λ and the new license Λ', defined as follows:

$$\Lambda' = [h(e_1, e_2, M, \eta', \delta, X)]_{dC1}$$

In step (4), C_2 verifies C_1's signature on the new license Λ' and validates η and M using the original Λ license. C_2 also makes sure that the η' license can be derived from η and also checks δ for the type of content being distributed. If all checks succeed, C_2 approves the transaction in step (5), sending C_1 a receipt φ, defined as:

$$\varphi = [h(e_{C1}, e_{CP}, [M]_{K'}, \delta, \eta']_{dC2}$$

The receipt φ represents an acknowledgement from C_2 that it received the content M with the rights η'.

C. Content distribution and relaying schema

In this section we provide a high level schematic of the content distribution architecture, as illustrated in Fig. 2:

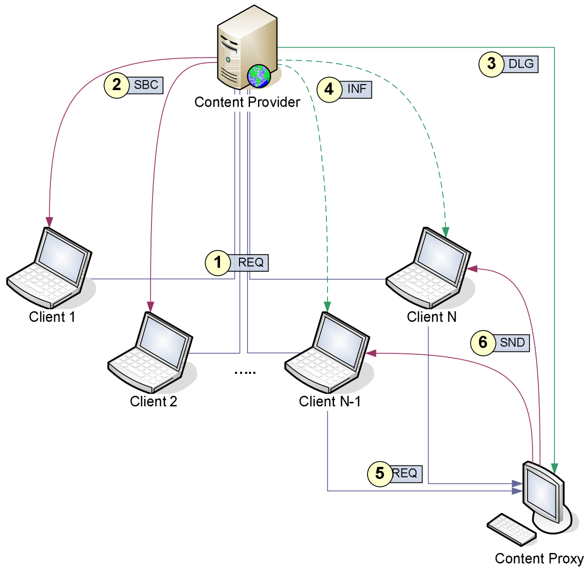

Fig. 2. Scalable distribution of digital content

1) A content provider *CP* receives, in a given time frame a number of *N* requests (marked with REQ in Fig. 2) for the same digital content. The CP will subject these requests to the approval of the other CPs which share the rights over that digital content, by using the secret splitting technique presented in Section III. If the requests are granted, *CP* will obtain a unique pair $<e_S, d_S>$ of session keys from the *MCP*.
2) The *CP* creates the sent-out message as described in the previous paragraph then sends it to all *N* clients who requested it, using the secure broadcast with secure lock method (SBC) [3] described in Section III. When a client receives the message, it will send an acknowledgement (ACK) message to the *CP*.
3) The *CP* checks if all *N* clients received the message. If there are clients who responded with a NAK (in our case clients *N-1* and *N*) or when the server is under high load, *CP* will send the original message to one or more registered content proxies, and delegate the task of distributing it to this proxy (step DLG in Fig. 2). If there is a registered proxy among the clients who successfully received the message, the CP will skip this step and go directly to step 4.
4) *CP* informs the clients which have not received the message that the proxy selected in step 3 contains a copy of the broadcasted message. From this point forward, *CP* will not be involved any more in the distribution of the digital content.
5) The clients notified by the *CP* in step 4 will request the broadcasted message from the content proxy. As stated in previous sections, the proxy stores the already encrypted version of the message in a transparent manner, and is not able to decrypt it if it is not among the legitimate receivers.
6) The proxy will forward the encrypted message to each of the requesting clients. If an illegitimate client requests this message, it will not be able to use it due to the protection provided by the secure broadcast protocol.

V. Threats

The proposed architecture introduces a number of threats, some of which being shared with existing DRM architecture and some being new. While a full discussion on the threats on DRM systems is beyond the scope of this paper, we will briefly outline them.

One of the most widespread threats is tampering with the compliant device or the tamper-resistant module inside. Good tamper-resistance is difficult to achieve [8, 9] so we can assume that security is effective against all but the most determined attacker.

Cryptographic techniques are hardly a threat today as attackers are smart enough to attack the weakest point. Content redistribution and collaboration between content providers are all target points for attackers and a whole new lot of attacks are possible:

- **Content masquerading** during the redistribution process. This may happen when a client receives a lesser value content than the one requested or a proxy client replaces the content to be further distributed.
- Since devices are susceptible to failure, they allow backing up the licenses and the content to unsecured media. This can lead to **untrusted storage backup attacks**.
- **Circumvented devices** are able to remove the security mechanisms that protect the digital content and therefore they can illegally distribute the content. Detecting and isolating circumvented devices is essential to the health of a DRM system.
- Because of the design if the content provider collaboration protocol, the **compromise of a single provider** leads to total failure of the content distribution. If a single content provider misbehaves it is very difficult to be excluded from the decisional process.
- Unlike other systems that deal with expensive operations (i.e. public key cryptography), our architecture is less susceptible to **denial of service attacks** as the task of distributing the content can be delegated the cooperating clients. DoS attacks are nonetheless possible so a number of ways to mitigate such threats are possible [10, 11].

VI. Conclusion and Future Work

Digital Rights Management systems are typically used by providers to restrict the ways consumers use the content. In this paper we proposed an architecture that matches close to real-world scenarios where several content providers share the right over a single piece of content. Our architecture is highly scalable as it enables simultaneous servicing of several clients that request the same content by using secure broadcasting and enables clients to become proxies in order to decrease the load on the central server.

Our architecture is also capable of preserving the rights of the distributed content even when redistributed from client to client, provided that certain requirements are met on the client device.

Currently we are working on adding resistance to denial of service attacks to the architecture and devising some guidelines for real-world implementation. We are also working on performance metrics of the architecture.

REFERENCES

[1] S. K. Nair, B. C. Popescu, C. Gamage, B. Crispo and A. S. Tanenbaum, “Enabling DRM – preserving Digital Content Redistribution”, 7th International IEEE Conference on E-Commerce Technology, 2005

[2] Bruce Schneier, “Applied Cryptography”, John Wiley & Sons, 1996, pag. 70

[3] G.-H. Chiou and W.-T. Chen, “Secure Broadcasting Using the Secure Lock”, *IEEE Trans. Software Eng.*, vol. 15, no. 8, pp. 929-934, August 1989.

[4] Trusted Computing Group, “Trusted Computing Platform Alliance Main Specification”, October 2003, Version 1.2, http://www.trustedcomputinggroup.org

[5] Eric W. Weisstein, “Chinese Remainder Theorem.” From *MathWorld* - A Wolfram Web Resource, http://mathworld.wolfram.com/ChineseRemainderTheorem.html

[6] eXtensible Access Control Markup Language (XACML), http://www.oasis-open.org/committees/xacml

[7] XrML: eXtensible Rights Markup Language, http://www.xrml.org

[8] R. Anderson, M. Kuhn, “Tamper Resistance – A Cautionary Note”, Proceedinggs of the 2nd Usenix Workshop on Electronic Commerce, pages 1-11, November 1996.

[9] Andrew Huang, “Keeping Secrets in Hardware: the Microsoft XBox™ Case Study, May 2002, http://web.mit.edu/bunnie/www/proj/anatak/AIM-2002-008.pdf

[10] Valer Bocan, “Threshold Puzzles. The Evolution of DoS-Resistant Authentication”, CONTI 2004 - Periodica Politehnica, Transaction on Automatic Control and Computer Science Vol. 49 (63), 2004, ISSN 1224-600X, http://www.dataman.ro/vbocan/download/conti2004-3.pdf

[11] Valer Bocan, Mihai Fagadar-Cosma, “Adaptive Threshold Puzzles”, EUROCON 2005 - The International Conference on "Computer as a tool", Belgrade, Serbia and Montenegro, http://www.dataman.ro/vbocan/download/EUROCON%202005%20-%20Adaptive%20Threshold%20Puzzles.pdf