SmartPro: A Smart Card Based Digital Content Protection for Professional Workflow

Alain Durand, Marc Éluard, Sylvain Lelievre, and Christophe Vincent

Thomson R&D France
Technology Group, Corporate Research, Security Laboratory
1 avenue de Belle Fontaine, 35576 Cesson-Sévigné Cedex, France
{alain.durand,marc.eluard,sylvain.lelievre,
christophe.vincent}@thomson.net

Abstract. This paper introduces SmartPro, a smart card based technology aiming at protecting content in professional workflows. It gives an overview on how SmartPro works. It also explains the design constrains that led to the use of smart cards and some of the extra difficulties implied by this choice in order to get to an implementation that may be industrially deployed.

1 Introduction

Digital Rights Management (DRM) has been for a decade a widely studied subject. 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). Piracy of digital content has actually been a growing issue since the entrance in the digital era and the widespread of high-speed communications. Black Market for DVDs are now important in most countries (see e.g. [1]). The MPAA (Motion Picture Association of America), the association of the seven major Hollywood studios, estimated to $6.1 billion the cost of video piracy in 2005 [2].

Generally, movie distribution obeys to different diffusion windows: film is first distributed in theaters, then in hotels or planes and DVD release occurs right after. It is then distributed to television first on Pay-Per-View or Video-on-Demand systems, then on Pay-TV and eventually on Free-To-Air channels. Table 1 shows average breakdown of movie revenues along the different diffusion windows.

Different solutions protect the release windows shown on Table 1. For instance, AACS [3] or CSS [4] protect home video / rental window while Conditional Access systems protect Pay TV or Cable TV window. DCI (Digital Cinema Initiative) specification [5] includes a protection scheme for digital theaters. Some systems (e.g., broadcast flag [6] or CPCM (Copy Protection and Content Management [7])) protect content distributed in the syndication window.

One could thus think that content protection technologies coverage is sufficient. This is however not the case. Roughly 10% of revenues loss is due to piracy operated before the content release in theaters [8]. This happened for instance to the recent Ridley Scott movie, American Gangster [9].

G. Grimaud and F.-X. Standaert (Eds.): CARDIS 2008, LNCS 5189, pp. 255–266, 2008.

Table 1. Movie Revenues (Courtesy of Technicolor)

	Theatrical Release	Airline / Hotel	Home Video / Rental	PPV / VOD inDemand, DirectTV	Pay TV HBO, Showtime	Networks Cable TV	Syndication
Time Frame (in month)	0	2-4	4-6 (ongoing)	6-9 (only for 30-45 days)	12-15	24-30	36-42
Typical/Approximate Revenues ($10M + box-office movie)	25%	1%	56%	2.5%	10%	3%	2.5%

The traditional approach to overcome this threat is based on network security technologies and on physical access control to facilities. While these techniques may be efficient against external attackers, this is not the case for insiders. This happened for instance for the third episode of Star Wars [10] that has been actually stolen from post-production facilities. It then passed through several go-betweens before being eventually made available on the Internet.

SmartPro technology is a content protection system for professional workflows. Its first real deployment aimed to prevent leakage from video production and post-production facilities. SmartPro is a smart card based technology that is for example deployed in Nexguard CP [11] product line. Smart card offers a secure place to store system keys and guarantees the integrity of the software using these keys. Any other transportable security token could be also used.

The main underlying idea behind this technology is to upgrade technologies or techniques that have been successful to protect the content in the consumer space to the professional realm. We propose to base the system on smart cards as Conditional Access systems do and we adapted the notion of consumer domain (see for instance [7]) to enable collaborative work.

The rest of the paper is organized as follows. In the next section, we give an overview of SmartPro technology. Next we explain how smart cards are used in SmartPro and give some design rationales. Finally, we present examples of difficulties when designing the system using secure processor.

2 General Presentation

SmartPro introduces the notion of Virtual Domain (VD). A VD is a set of devices that can share private contents. It can represent a company or part of a company like a post production facility. A VD is not bound to a person or a physical location. It can be used for any content format and using any network technology or physical interface.

Inside a VD, content is scrambled, i.e. encrypted, with a cryptographic key[1]. This key is protected so that only devices belonging to the VD can access to

[1] In the video content industry, the term scrambling is typically used rather than encryption for content protection. This term make references to the first mechanisms for Pay-TV where some parts of the content where re-ordered to achieve protection.

it, and thus to the content. SmartPro mainly brings a key management system implementing this notion of Virtual Domain.

2.1 Actors in Virtual Domain

Figure 1 illustrates the basic elements of a VD. Acquisition devices are the entry points of the VD. From that point on, the content is digital and protected. It can be accessed, if allowed, by any renderer devices, or processing devices of the VD.

Renderer devices are the final points of the VD. After a renderer device, content does not benefit anymore from SmartPro protection. Special care will be needed within the renderer devices to avoid theft of content once it is no longer SmartPro protected. Content may be in the clear or protected by another copy protection scheme (for example with watermark).

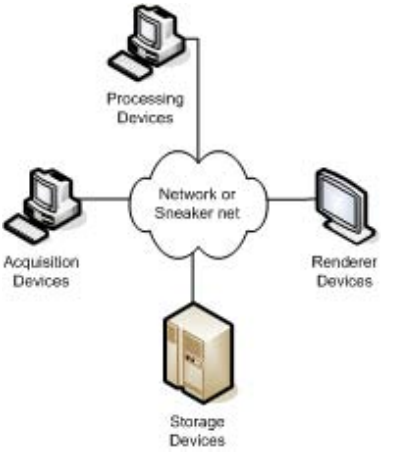

Fig. 1. Devices in a Virtual Domain

Processing devices modify SmartPro content. Prior to applying the expected process, device unprotects the content or part of it. Once processed, the device re-protects the content. All these operations are performed in a secure environment (i.e. content remains in the same VD). SmartPro processing devices cannot create SmartPro protected content from clear content.

Storage devices do not act on the content at all. They are simple bit buckets.

2.2 Content Protection

SmartPro protected content is always scrambled. Scrambling mechanism is based on robust cryptographic algorithm (AES-CTR [12]). The scrambling keys are called Control Words (CW). Usage and access rules of the SmartPro protected content are called Usage Rights. Control Words and usage rights are

embedded in licenses called Local Enforcement Copy Management (LECM). A given license is valid only in one Virtual Domain for only one content. License is created in the acquisition device at the same time as the content is acquired. Licenses are analyzed and enforced by devices before they handle the content. A processing device can modify the license if granted in the usage rights.

License is partially encrypted with a secret key called Domain Key (to ensure confidentiality of CW) and signed (to ensure integrity of LECM). This key is unique to a VD and is randomly generated at VD creation. All devices belonging to a given VD share its Domain Key. A device belongs to only one VD.

2.3 Multiple Virtual Domains

A simple VD may not be sufficient for many cases. For instance, two firms *A* and *B* may use SmartPro. Each manages its own VD. In some cases, some protected data may be transfered from domain *A* to domain *B* but data must remain protected during this transfer. Thus, SmartPro introduces the notion of Multiple VDs. Content is able to flow through controlled devices, called Bridge devices, from one VD to another VD. The owner of the source VD manages the usage rights of the delivered content to the destination VD. Figure 2 illustrates this architecture.

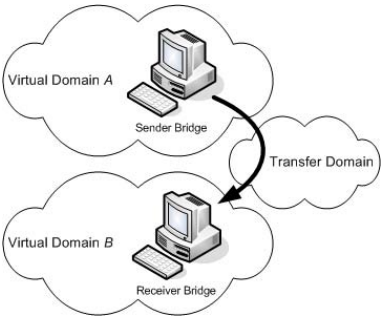

Fig. 2. Example of multiple Virtual Domains

The bridging operation between two VDs is performed with two entities: The Sender Bridge prepares and sends a content from source domain into a transfer domain. The Receiver Bridge receives and transfers a content from the transfer domain to the destination domain. This transfer domain is temporary and completely transparent for the user. Content remains unchanged (i.e. scrambled) during the operation and only the licenses are processed. Usage Rights associated to the content may be modified by the Sender Bridge.

Sender Bridge is able to send content to multiple VDs. Receiver Bridge is only able to send back the content to the source domain.

3 Device: A Collaboration between Host and Token

3.1 Overview

Since no software solution can be considered sufficiently secure, the management of sensitive operations and secret data should be done by secure hardware. We choose Smart cards (called tokens) as secure hardware to store secret data and run applications in a tamper resistant environment. Nevertheless, their processors do not permit to encrypt or decrypt large amount of data in a reasonable time.

Thus, it is preferable to use a host with a powerful processor associated to a token. The content is scrambled or descrambled by the host while the license is managed by the token.

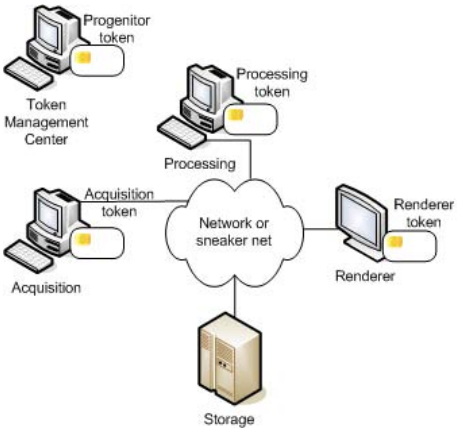

Fig. 3. Architecture of Virtual Domain with tokens

To keep the domain key as secure as possible, it is managed by the token and never leaves it. However the control words that protect content need to be used by tokens and also by hosts. If an attack succeeds on a host, only control word may leak and so only the corresponding content may be broken. The other contents of the domain remain secure. If an attacker targets all contents of a domain, he must extract the domain key from the token.

The host is not bound to any domain until a token is inserted. The host is then temporary bound to the token's domain. So, only the tokens belong to a Virtual Domain.

3.2 Token Management Center

Token Management Center (TMC) has a major role in key management. It performs the enrollment or activation of tokens in a Virtual Domain. A special token, the progenitor token, is associated to each Virtual Domain. There is only one progenitor in a Virtual Domain. The activation of a new token to a VD requires the presence of its progenitor token in the TMC.

TMC runs on standard computers. It supports simultaneously at least two tokens. The TMC does not store any secret. The progenitor token handles all the secrets including the domain key.

All progenitor token are delivered inactive to the user and activated using TMC. During this activation, the progenitor generates a domain key and securely stores it. During the activation of a token, the progenitor securely transfers the domain key to the token.

SmartPro supports revocation of domains, tokens and hosts (see section 4.1).

The progenitor token creates and maintains a database to store the serial numbers of the tokens that has been activated or revoked in the domain.

The TMC does not need to be online with any device of the VD, or any back-office. Nevertheless, online connection with devices may allow remote management of tokens.

3.3 Hosts and Tokens Interactions

Once the token is linked to a domain, it can deal with SmartPro content. In an acquisition device, the host receives clear content to be scrambled, it requests the token to build a license. The token picks up a random control word. It inserts the CW in the license and encrypts the license with the domain key. Then, token sends both CW and license to the host. The host scrambles the content with the CW.

To descramble a protected content, a renderer device needs the license corresponding to the content. The host sends the license to the token. If the token and the content are in the same domain, the license can be decrypted using the domain key. Then, the decrypted CW can be sent to the host to descramble the content.

The domain key never leaves the token, only CW is provided to the host. The token shall first ensure that the host is trustful, and compliant. Non-Compliant hosts could divulgate the CW. A compliant host, even purely software, protects the CW and the content upon its descrambling. Hence, some secure coding techniques (e.g.: code obfuscation, anti-debugger) are used to make difficult to modify host behavior. Furthermore, before sending the CW to the host, the token authenticates the host. To that end, the host needs to have a public/private key pair. Finally, the CW needs to be sent encrypted since communications between the token and the host are easy to eavesdrop. For these reasons, a Secure Authenticated Channel (SAC) is setup prior to any communication.

3.4 Secure Authenticated Channel

The SAC is used by progenitor token during token activation or token deactivation and for CW transmission to the host (see section 3.3).

The protocol is based on Diffie-Hellman [13]. Each entity is given random private key (K_{priv}) and a certificate that embeds an identity (e.g., the certificate serial number) and the public key ($K_{pub} = g^{K_{priv}} \mod p$ where g and p are Diffie-Hellmann parameters shared by all entities). For the sake of simplicity, the notation mod p will be omitted in the rest of the document but it shall be understood that all exponentiations of g are performed modulo p.

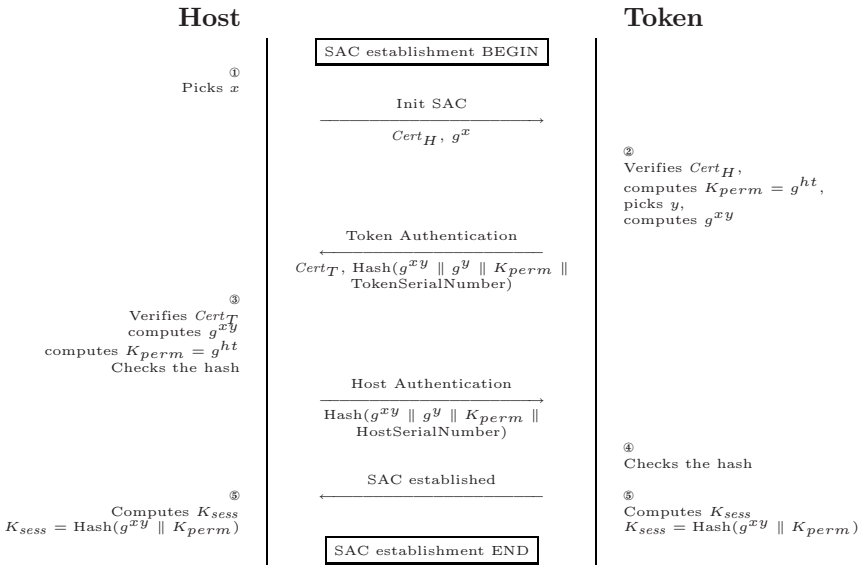

Fig. 4. Secure Authenticated Channel Protocol

1. The host picks a random x, computes the associated public value g^x and sends the result to the token together with its certificate $Cert_H$.
2. The token extracts the public key g^h from the host certificate. It verifies that the certificate is valid. It then computes secret key $K_{perm} = g^{ht}$ where t is the token certificate secret key. The token also picks a random y, computes the associated public value g^y. It computes as well the hash value of the concatenation of g^y, g^{xy}, K_{perm} and its serial number. It sends the result of both computations together with its certificate to the host.
3. The host extracts the public key g^t and verifies that the certificate is valid. It then computes secret key $K_{perm} = g^{ht}$ and g^{xy}. It also checks whether the received hash value is correct. It computes then the hash value of the concatenation of g^x, g^{xy}, K_{perm} and its serial number and sends the result to the token.
4. The token verifies the correctness of received hash value.
5. Both token and host compute the session key K_{sess} as the hash value of g^{xy} and K_{perm}. K_{sess} will be used to secure further communication between the host and the token.

4 Using Secure Processor

4.1 Revocation Mechanism

The security of SmartPro is based on a removable secure processor (a token). It guarantees that all the secret data are securely stored and processed.

Nevertheless, we know that no security system is 100% secure. Hackers use more and more sophisticated tools that will eventually defeat any security mechanism. Thus, it is important to have a revocation mechanism that will prevent a compromised element from working. In case of major hack, the replacement of all tokens should be planned.

The revocation mechanism defined in the SmartPro specification is based on two revocation lists:

- The Internal Revocation List (IRL) contains the elements revoked in a given domain and is managed (created and updated) by the domain manager (progenitor token). The IRL only addresses token elements and contains the Certificate Serial Number (SN) of the revoked tokens.
- The External Revocation List (ERL) contains the elements revoked in all SmartPro system. The ERL addresses token, host and VD elements. The ERL contains serial number of host and token, and Virtual Domain identifier (VDID) for Virtual Domain.

Revocation List Usage. During the first messages of the SAC establishment, the host and the token exchange their certificate. At this moment, each entity checks that the serial number of their peers certificate is not present in the revocation list. If so, the SAC establishment continues as specified in Section 3.4.

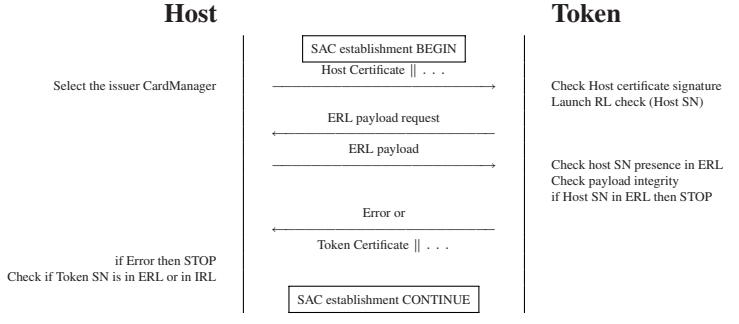

Fig. 5. Revocation List Usage Protocol

The difficulties to implement our revocation mechanism were:

- The token has a given limited amount of memory. It cannot store an ever increasing list.
- The token and the host must always hold the same version of the lists.

Revocation List Format. Each list has a header and a payload. The header contains an ever increasing index of the list, and for each element type (host, token or VD), the number of revoked elements and a digest (SHA-1) of the list of revoked elements. The header is signed by a root revocation key for the ERL and the progenitor revocation key for the IRL. The payload gives for each element type, the SN (or VDID for virtual domain) of the revoked elements.

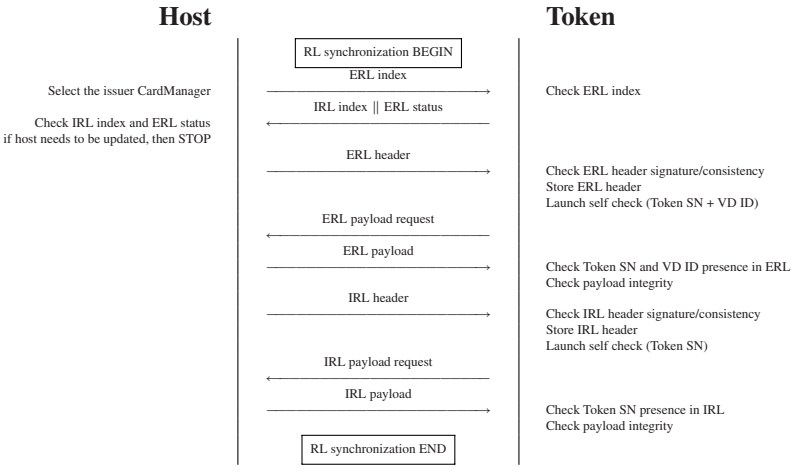

Fig. 6. Revocation List Synchronization Protocol

The lists indexes have been integrated in the messages exchanged between the host and the token during the SAC establishment. Thus, the SAC will not be established if the host and the token are not synchronized on the same lists.

Revocation List Storage. The token only stores the list headers. When it receives its host certificates (during the SAC establishment), it requests the list to the host prior to checking if the certificate is in the list. It checks the validity of the received list using the digest value contained in the header.

Revocation List Synchronization. The host sends to the token its external RL index and the token responds with its internal RL index and an External Status indicating if an update is needed. If the token needs an update, the host sends the new RL header. If the host needs an update, it must retrieve the new RL before any further collaboration with the token.

4.2 Bridging Implementation

Another key issue in our implementation was the bridging mechanism. It transfers content from a source domain to one or several other destination domains. One solution would be to send the license of the domain source to the relay token which would decrypt it and re-encrypt it for each destination domain. This means that the relay token should contain secret keys of all the potential destination domains!

Our solution uses two kinds of host/token:

– The Master Bridge (MB, host and token): It only knows source Domain Key. It converts the license for the source domain into a license for a transfer domain. It then generates descrambling information specific for each destination domains (CDI for Content Descrambling Information). The Master

Bridge token is initialized in the source domain. It holds a certificate containing its serial number (MB SN).
- The Simple Bridge (SB, host and token): It processes the license from the transfer domain and generates a license for a destination domain. A Simple Bridge token is initialized twice. First it is activated in the destination domain. Then it must be registered in the source domain: It receives the information needed to process the CDI generated by the Master Bridge token. These information include an initialization index. This index is ever increasing in the source domain and is used by the Master Bridge token to generate the CDI. The Simple Bridge holds a certificate containing its serial number (SB SN).

The source domain progenitor generates and manages the following elements:

- A master key for Master bridge (MK_{MB}) used to calculate derived key for Master bridge (DK_{MB}): $DK_{MB} = E\{MK_{MB}\}(MB\ SN)$.
- A master key for Simple Bridge (MK_{SB}) used to calculate derived Key for Simple Bridge (DK_{SB}): $DK_{SB} = E\{MK_{SB}\}(SB\ SN \parallel SB\ index)$.
- The Authorization Mask is a bit mask where the position of each bit corresponds to a SB index. If the bit is "1", the corresponding SB is registered and not revoked in the IRL of the source domain. The Authorization Mask is generated and updated by the progenitor and signed with the progenitor revocation private key. The Authorization Mask is transmitted to all Master Bridge hosts. The Authorization Mask will be used by the Master Bridge to know if it can generate CDI for a given Simple Bridge. The use of Authorization Mask simplifies the operation in the Master bridge token: all checks relative to Simple Bridge registration or revocation are performed by the progenitor.

The Master Bridge token holds the following information received from the progenitor during its activation:

- MK_{SB},
- DK_{MB} calculated by the progenitor.

The Master Bridge token also stores the latest version of the Authorization Mask. The update of the Authorization Mask is performed by the Master Bridge host before any bridging operation in order to take into account new registered or revoked Simple Bridge tokens.

The Simple Bridge token holds the following information received from the progenitor during its registration:

- MK_{MB},
- DK_{SB} calculated by the progenitor,
- an initialization index.

On the Master Bridge side During a bridging operation, the Master Bridge host sends the license of the content to its token. The license is converted into a

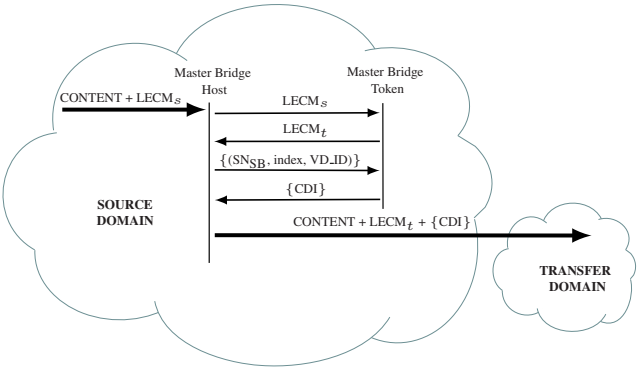

Fig. 7. Master Bridge

license for a transfer domain, encrypted by a transfer license key (TK_{LECM}). Then, for each destination domain, information on the Simple Bridge is sent to the Master Bridge token (SB SN, SB index and VD_ID). The Master Bridge token creates CDI. A CDI contains TK_{LECM} encrypted with a bridge key (BK). The Bridge Key is calculated from DK_{MB} and DK_{SB}: $BK = DK_{MB} \parallel DK_{SB}$. The Master Bridge token calculates DK_{SB} with MK_{SB} and the Simple Bridge token information. The CDI also contains the serial number of the targeted Simple Bridge token. The scrambled content, the transfer license and the list of CDIs are transmitted to each SB device.

On the Simple Bridge Side When receiving these data, the Simple Bridge host sends the license and its CDI to the Simple Bridge token. The token calculates DK_{MB} with MK_{MB} and the Master Bridge token information. It can then calculate the bridge key and retrieve TK_{LECM}. It then converts the transfer license into a license for its domain.

The content is not processed and remains scrambled during the bridging operation.

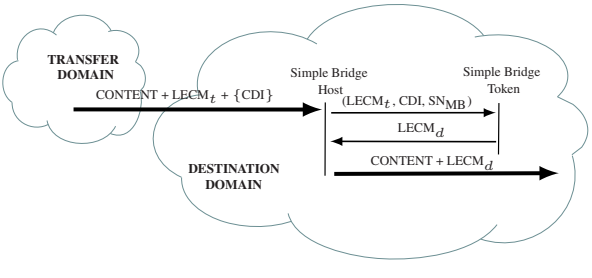

Fig. 8. Simple Bridge

5 Conclusion

SmartPro is a content protection scheme preventing content leakage in professional workflows. A main design criteria was to achieve high security and renewability. Hence, the choice of a smart card based implementation was straightforward.

This choice however led to a greater system complexity due to lack of computational power and bandwidth of smart cards. We had to design original mechanisms to deal with flexible cards and hosts revocation.

SmartPro only cares about basic layers of content protection. Other techniques may be plugged in the upper layers. For example, Nexguard Content Protection [11] uses watermarking technology allowing to trace back the origin of the content leakage. Adding a rights expression language or an access control technology would allow to further control the distribution of the protected content, e.g., by setting a content license expiration time or a user-based access granularity. Whatever the technology plugged above SmartPro is, the system designer will face the same constraints to use adequately the protection offered by the smart card without impeding the whole system.

References

1. Fake DVD seizures up 41% on 2004(last visited May 2008), http://news.bbc.co.uk/1/hi/entertainment/film/4099696.stm
2. MPA 2005 US piracy fact sheet (last visited Febuary 2008), http://www.mpaa.org/USPiracyFactSheet.pdf
3. Advanced Access Content System for Pre-recorded Book (AACS), v0.91, February 17, 2006 (last visited Febuary 2008), http://www.aacsla.com/specifications/
4. CSS Description (last visited Febuary 2008), http://en.wikipedia.org/wiki/Content_Scramble_System
5. DCI Specifications (last visited Febuary 2008), http://www.dcimovies.com/specification/index.tt2
6. Broadcast Flag Description (last visited Febuary 2008), http://en.wikipedia.org/wiki/Broadcast_flag
7. CPCM Description (last visited Febuary 2008), http://www.dvb.org/technology/dvb-cpcm/
8. Byers, S., Cranor, L., Cronin, E., Kormann, D., McDaniel, P.: Analysis of security vulnerabilities in the movie production and distribution process. In: ACM workshop on Digital rights management, October 27 (2003)
9. http://o.seattletimes.nwsource.com/html/movies/2004016889_gangster16.html (last visited Febuary 2008)
10. http://news.bbc.co.uk/1/hi/entertainment/4650956.stm (last visited Febuary 2008)
11. NexGuard Content Protection (last visited Febuary 2008), http://www.thomson.net/GlobalEnglish/Products/content-tracking-and-security/nexguard/nexguard-content-protection/Pages/default.aspx
12. Daemen, J., Rijmen, V.: The design of Rijndael: AES – the Advanced Encryption Standard. Springer, Heidelberg (2002)
13. Diffie, W., Hellman, M.E.: New Directions in Cryptography. IEEE Transactions on Information Theory IT-22(6), 644–654 (1976)