A PROTECTION PROCESSOR FOR MPEG-21 PLAYERS

Paolo Nesi, Davide Rogai, Andrea Vallotti
DSI-DISIT, Department of Systems and Informatics, University of Florence
Via S. Marta, 3 - 50139 Florence, Italy
nesi@dsi.unifi.it, http://www.disit.dsi.unifi.it

ABSTRACT
The design and implementation of MPEG-21 players and authoring tools presents several critical points to be solved. One of the most relevant is the security level and protection processing in the players. This paper presents a solution for the realization of components in charge of enforcing Digital Rights Management in AXMEDIS tools for MPEG-21 digital content. The proposed architecture provides functionalities to create both trusted environment on the client side and dynamic protection and unprotection of digital content including digital resources and their organization and metadata. The same solution can be used to achieve the desired security level in any other MPEG-21 player or authoring tool. The architecture presented hereinafter has been adopted to enforce protection on authoring and player tools developed for the AXMEDIS IST FP6 R&D European Commission project.

1. INTRODUCTION
Several requirements about content protection have to be met in order to allow the digital management of rights for business models based on digital content commerce, Specific DRM (Digital Rights Management) technologies and solutions have to be enforced to ensure that the protected content is only consumed by the authorized customers, while revenues are returned to each and every stakeholder along the content value chain. The present DRM solutions typically address single channel distribution and they are not interoperable in terms of DRM; this means that the interchange of content is possible among tools only if they have the same protection model and license. The new frontiers of DRM interoperability would see the content moving among distribution channels and tools based on different DRM license models (MPEG-21, ODRL, XRML, etc.). In addition, the digital resources contained in the distribution packages (such as: MPEG-21 DI, OpenSky, etc.) may be protected by using different algorithms and models, thus describing the so called IPMP information (Intellectually Property Management and Protection information).

A quite general content consumption scenario is summarized in Fig. 1. The content is distributed (e.g., by means of different distribution channels) through a protected content package. When the content is used or played, it resides (as a segment or totally) in the customer's devices. The device for content usage has to be authenticated and authorized. Secret information, like the key, has to be delivered to content usage tools in order to unprotect the content. To avoid unauthorized content usage, the consumer has to be kept unaware of the necessary secret information.. The secret information is provided only to authorized consumption tools on the basis of some licenses.

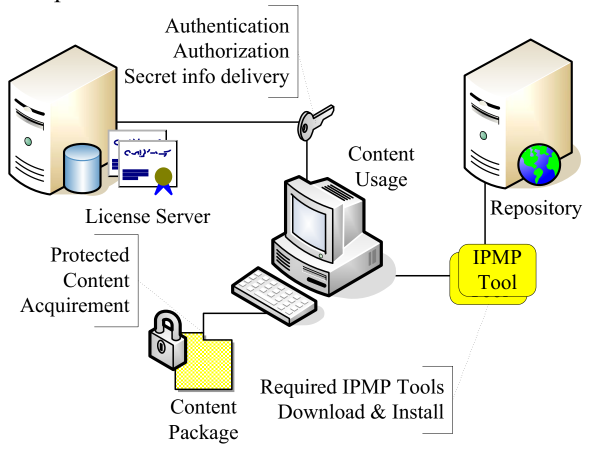

Figure 1 – Scenario on intellectual property protection

Cracking consumption tools is a typical process which aims at getting clear content resources or secret information, so as to produce the former. In most of the current protection solutions, the protection model (tools and algorithms to unprotect the content) and, in some cases, the secure information (e.g. keys), are unique for every distributed content. These solutions are intrinsically unsafe, since once the model is cracked, the hacker may unprotected all the digital resources, an example is found in the encryption used for DVDs.

The system should not be founded upon a single content consumption tool, on the contrary it should allow introducing new tools according to a preventive behavior certification. The system should merely verify whether a certified tool has either been cracked or not, in order to prevent any malicious behavior. A solution where the protection tools are dynamically replaced is necessary to strengthen content security In this way, content producers can distribute their packages by exploiting the best protection tools, while loss of control on large amounts of valuable content due to a unique crack action and effort is easily prevented. The content usage tool should be up to getting dynamically each specific protection tool needed to unprotect a given content package. A repository for downloading protection tools should be accessible.

MPEG-21 standard is working on the standardization of Intellectual Property Management and Protection, IPMP, which aims at defining how to represent the protected content in packages and the required information for unprotecting content in an interoperable manner [1].

In AXMEDIS, the content package is the AXMEDIS Object. It is an extension of the MPEG-21 Digital Item [2]. In AXMEDIS, the MPEG-21 IPMP has been used, but unlike MPEG-21 IPMP, which is limited to model protection information representation, the AXMEDIS has studied, defined and realized the whole distribution and consumption architecture by introducing specific services for: (i) authorization, (ii) authentication and certification, and (iii) processing of protection information in several different conditions, e.g., streaming, download, etc.

This paper presents relevant technologies introduced by AXMEDIS in order to get the consumer devices more reliable and to exploit actual content under DRM. AXMEDIS Protection Processor is the component being responsible for (i) device and tool authentication and certification, and for (ii) processing of protection information. It realizes the DRM enforcement by actually unprotecting content for authorized consumption and by verifying device integrity and protection.

This paper is organized as follows. Section 2 offers an overview of the parts being interested in MPEG-21. Section 3 outlines the AXMEDIS Protection Processor. Section 4 shows how the consumer device reliability is achieved. Section 5 focuses on dynamic content protection and unprotection details. Conclusions are drawn in section 6.

2. MPEG-21 FEATURES

MPEG-21 is mainly focused on the standardization of formats related to the e-commerce of digital objects and DRM aspects. In particular, MPEG-21 scope is mainly related to content formats, leaving completely aside the definition of system architectures, business models, etc. The standardization process of MPEG-21 is still under completion [3]. At present many parts are mature, whereas other ones are still under evolution. The quite stable parts of the standard, being most relevant to this work, are:

- Digital Item Declaration (DID) defines how Digital Items (DIs) have to be represented. A DI is a structured digital object and it is the fundamental unit of distribution and transaction within the MPEG-21 framework. A DI is a wrapper for digital resources and related metadata [2];
- Rights Expression Language (REL) provides means to formalize licenses in a machine-readable way. A license grants to a user the rights to act somehow on a given object and/or digital resource contained;
- Intellectual Property Management and Protection (IPMP) allows to include protected content inside Digital Items. Any Digital Item element has a corresponding protected representation. Every IPMP element has to include protection information to enable its unprotection from different MPEG-21 compliant tools [1].

A DI is represented as an XML document which fulfils the DI Declaration Language (DIDL) schema. DIDL has been extended by IPMP to include representation of protected elements. IPMP defines an XML schema to describe the needed components (identified by a unique identifier), their usage order, how to retrieve them, etc.

Please note that, the MPEG-21 standard does not define anything regarding (i) the description of initialization settings, (ii) the solution for authentication and certification of devices and tools, (iii) the processing of protection information.

3. PROTECTION PROCESSOR

In the AXMEDIS architecture, every tool which is able to manipulate content, is developed on the basis of the AXMEDIS Object Manager (AXOM). This module can manipulate MPEG-21 DIs and/or AXMEDIS Objects. The Protection Processor is integrated within the AXOM in order to both ensure the reliability of the latter and allow the management of protected objects as well [4], [5]. Any AXMEDIS tool can be verified against reliability. The AXMEDIS Protection Processor is the module, charged with the task of coping with the requested security level in the AXMEDIS tools. In further details, the problems related with client side authentication and certification and dynamic unprotection of content are addressed. These aspects are fundamental to enable the enforcement of DRM. For this purpose two different technologies and components have been created: the estimation and processing of Device Fingerprint and the dynamic protection/unprotection engine.

4. DEVICE FINGERPRINT

How can content owners/distributors be sure that their content is correctly used once it has been stored on the client device? As to the client device, what is meant here is the couple content tool and platform. The device could be tampered, e.g., allowing the final user to save content in its unprotected form on a storage device. The only way to avoid this is having the device continuously “controlled” by an external certified authority, which should be able to detect tampering actions. The control mechanism cannot be pervasive in the consumer system, however, it is feasible to suppose that it can gather some information in order to verify periodically some integrity properties.

The main issue is to define which data have to be collected from the client device, in order to guarantee high robustness against malicious users. Moreover, it is worth to define a set of data which “uniquely” identifies a device. This set of data can be referred to, as the Device Fingerprint. Furthermore, it is desirable to create a detection policy in order to prevent typical device maintenance activities from being considered as malicious behaviours. The solution has to allow detecting only the actual violations of the device reliability. In that way, black-lists of tampered devices can be created and managed, i.e., once a tampered device has been detected, it cannot be allowed to use protected content anymore. At the next verification such device is disabled or the grant needed to perform the content access is denied.

In order to arrange a set of data which satisfies the above requirements, uniqueness of the data and robustness (against tampering) of retrieving methods are key points that have to be taken into account. In fact, the uniqueness of the retrieved data guarantees the uniqueness of the Device Fingerprint. Moreover, the more robust the retrieving methods against attempts of tampering are, the more difficult and detectable such attempts become. The proposed fingerprint has been designed considering both platform (i.e., OS, peripherals, storage, etc.) and tool features (properties about installed component). In fact, fingerprint data allow identifying the device and detecting possible malicious modifications to the installed components. In details, the proposed fingerprint is composed by the following data:

- for each hard-disk: serial, description
- for each processor: serial, name, description, vendor
- BIOS: serial, name, version
- for each network device: MAC address, name
- operative system: name, version, installed upgrade (e.g., SP1), serial (e.g., product id)
- For each relevant component/resource file: full name (path and file name), physical position, digest (e.g. SHA1), creation date and time, last modification date and time, size, etc.
- An alphanumerical string that identifies the tool type (e.g., ACME's authoring tool, BrandX's player, etc.)

Files containing executable code and managing directly clear-text content (e.g., rendering) are considered relevant because they can be modified in order to obtain illegally the unprotected content. Moreover, some configuration files can be considered relevant. They can only be controlled on the basis of their physical position and full name, since it has no sense to control digest of files which may vary. However, such controls are useful to detect unexpected movements of such files. The proposed fingerprint data cannot be completely estimated on all kind of devices (e.g., a Pocket PC does not have any hard disk), however, they cope with all the most important features of digital system, thus assuring good result in all context. Furthermore, the fingerprint includes several unique identifiers (e.g., serial numbers) and it can assure a good level of uniqueness, even if not all the data are available. Table 1 summarizes those tampering attempts which can be detected by using the proposed fingerprint.

As to enabling the realization of a unique detection system for any type of digital devices, the proposed fingerprint data have to be arranged in a common format. To achieve this goal, an XML schema has been designed to carry out this information [4].

Tampering activity	**Detection capability**
Modifying executable components	Component properties and digests change w.r.t. tool registration
Moving/substituting relevant files	File properties (e.g., physical position, size, etc.) change
Tool Copying on other devices without registration	Disks, processors and OS serials change
Modifying platform firmware	BIOS properties change

Table 1 – Some attacks and detect capabilities

Since the fingerprint has been analyzed and developed to be applicable to any kind of device, it is also worthwhile realizing a software module in order to estimate a fingerprint as portable as possible. Currently the interface described above has been implemented for two platforms: Windows OS and Linux OS both on Intel platform. The methods, which have been used in these two implementations, have been also analyzed w.r.t. their robustness against tampering attempts. This means that what has been checked is how much difficult to fake their result has become. The proposed approach has been compared with the one used in some Microsoft products: Windows Product Activation (WPA).

5. CONTENT PROTECTION/UNPROTECTION

As stated in the introduction, a key point for the protection of the intellectual property is the controlled access to digital resources. Usually protection means the ciphering of a resource with some kind of encryption algorithm. However, resources could be protected in other ways, e.g., scrambled, compressed, combining different algorithms, therefore a generic encoding/decoding process has to be considered.

In order to deal with this variability, what is necessary is a common way to describe the protection information (i.e., how to get the decoded resource). In addition, it is worthwhile managing different protection components in a uniform way.. In fact, on the basis of the protection information the device should be able to retrieve automatically and orchestrate the described components in order to perform the needed decoding steps, thus obtaining the original digital resource.

As to the common representation of the protection information, the MPEG-21 IPMP standard has been chosen as our research work basis. As stated in section 2, the standard does not define anything regarding the description of initialization settings. This lack of standardization could cause the proliferation of several description models for such data, thus forcing to include in each protection component the knowledge to parse specific initialization settings (e.g., including an XML parser). The proposed solution, modeled for AXMEDIS tools, prevents from redundancy of parsing capabilities among different components, by providing a generic XML schema. In this way, the parsing capabilities of initialization settings can be included in the content usage tools which exploit functionalities exposed by protection components. This schema allows defining parameters as a triplet: name, type, value. Basic types (such as integer, float and string) have been considered. The schema is generic enough to address any suitable initialization settings and, at the same time, it can be easily parsed.

Another major issue about protecting/unprotecting content is to define a common interface for protection components and a unique way to manage them while considering that resources should be decoded and unprotected in real-time without saving temporary non-protected information on some storage. This requirement is essential in order to reach a good security level. Moreover, decoding is much more delicate than encoding, since usually the former is performed on the consumer device. However, for the sake of simplicity, the same model should be used for both.

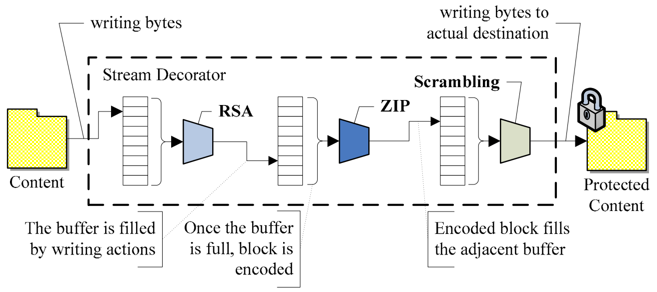
writing bytes
writing bytes to actual destination
Stream Decorator
RSA
ZIP
Scrambling
Content
Protected Content
The buffer is filled by writing actions
Once the buffer is full, block is encoded
Encoded block fills the adjacent buffer

Figure 2 – An example of a Protection stream

In order to satisfy the above requirements, a stream approach has been adopted. That is, the proposed solution handles a resource (to be encoded or decoded) as a stream of bytes regardless from its real nature. Resource encoding and decoding have been respectively modeled as writing and reading actions performed on streams. The proposed solution allows building a chain of encoding algorithms. An example is shown in Fig. 2: the original resource can be protected by simply writing to the output stream which is placed at the beginning of the protection chain. More specifically, the output stream is a Decorator [6] of the actual destination stream. Symmetrically, an encoded resource can be decoded by reading from the input stream placed at the beginning of the chain. This stream is a Decorator of the actual input stream that gets data from the protected resource. While performing a writing (reading) operation the resource data flow is processed by all the encoding (decoding) algorithms in the chain, before being actually written to (read from) the destination (source). The decoding chain has to be created on the basis of the protection information of a given protected resource. The protection information to be processed can model chains which include tools addressing partial content segments. Furthermore, a given protection tool can be reused more than once with different initialization settings, parameters.

Each protection component has been schematized as an algorithm, which gets maximum N bytes as input and produce maximum M bytes as output. The algorithm itself can do any kind of elaboration on the input bytes therefore the component itself can contain any state-of-the-art technology, e.g. an RSA encryption, a ZIP compression, etc. These algorithms are interconnected one another through buffers allowing to manage the differences between block lengths of adjacent algorithms. These buffers are set up while initializing the chain. Additional problems while enabling typical player actions, such as fast play, go forward and backward, are addressed by this architecture as well.

6. CONCLUSIONS

A flexible and extendable model to address content protection, considering also MPEG-21 IPMP has been designed and implemented. It includes classes and infrastructure to check device integrity and to dynamically protect/unprotect content by applying different algorithms, thus achieving trustiness on content consumption under DRM. An additional modeling to provide a uniform manner to specify protection component settings has been created. This has allowed a simpler interaction with the object model. To retrieve additional details on the presented work, please refer to deliverables [4] of the AXMEDIS project [7]. Both device fingerprint and dynamic protection/unprotection mechanisms have been validated. The former has been evaluated to be robust enough against typical tampering actions of executable files. The latter has been used in the AXMEDIS Editor in order to perform protection and unprotection of authored content packages.

7. ACKNOWLEDGMENTS

The authors would like to thank all the AXMEDIS project partners such as TISCALI, HP, EUTELSAT, DSI, FUPF, SEJER, GIUNTI, FHG, Univ. Leeds, etc., for their contribution, and collaboration. A specific acknowledgment to the EC IST FP6 for partially funding the AXMEDIS project.

8. REFERENCES

[1] MPEG Group, “Introducing MPEG-21 IPMP Components”, www.chiariglione.org/mpeg/technologies/mp21-ipmp/, December 2005
[2] MPEG Group, “Introducing MPEG-21 DID”, www.chiariglione.org/mpeg/technologies/mp21-did/, December 2005
[3] L. Chiariglione, MPEG Group, “The MPEG Home Page”, www.chiariglione.org/mpeg, December 2005
[4] AXMEDIS DE3.1.2A “Framework and Tools Specifications”, www.axmedis.org/documenti/view_documenti.php?doc_id=1379, December 2005
[5] P. Bellini, P. Nesi, “An Architecture of Automating Production of Cross Media Content for Multi-channel Distribution”, in Proc. Axmedis 2005, Florence, Italy, Nov 2005, IEEE Press.
[6] E. Gamma, R. Helm, R. Johnson, J.Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, 1995.
[7] AXMEDIS web site. http://www.axmedis.org/, December 2005