MPEG-4 IPMP Extension for Interoperable Protection of Multimedia Content

**Ming Ji,[1] S. M. Shen,[1] Wenjun Zeng,[2] Taka Senoh,[3] Takafumi Ueno,[3]
Terumasa Aoki,[4] Yasuda Hiroshi,[4] Takuyo Kogure[4]**

[1] *Panasonic Singapore Laboratories Pte Ltd., Block 1022 Tai Seng Avenue #04-3530, Tai Seng Industrial Estate, Singapore 534415*
Emails: mji@psl.com.sg, shen@psl.com.sg

[2] *Department of Computer Science, University of Missouri-Columbia, MO 65211, USA*
Email: zengw@missouri.edu

[3] *Matsushita Electric Industrial Co., Ltd., Osaka 571-8501, Japan*
Emails: senoh.taka@jp.panasonic.com, ueno.takafumi@jp.panasonic.com

[4] *Center for Collaborative Research (CCR), Research Center for Advanced Sciences and Technology, University of Tokyo,*
4-6-1 Komaba Meguroku, Tokyo 153-8904, Japan
Emails: aoki@mpeg.rcast.u-tokyo.ac.jp, yasuda@mpeg.rcast.u-tokyo.ac.jp, kogurent@attglobal.net

To ensure secure content delivery, the Motion Picture Experts Group (MPEG) has dedicated significant effort to the digital rights management (DRM) issues. MPEG is now moving from defining only hooks to proprietary systems (e.g., in MPEG-2, MPEG-4 Version 1) to specifying a more encompassing standard in intellectual property management and protection (IPMP). MPEG feels that this is necessary in order to achieve MPEG's most important goal: interoperability. The design of the IPMP Extension framework also considers the complexity of the MPEG-4 standard and the diversity of its applications. This architecture leaves the details of the design of IPMP tools in the hands of applications developers, while ensuring the maximum flexibility and security. This paper first briefly describes the background of the development of the MPEG-4 IPMP Extension. It then presents an overview of the MPEG-4 IPMP Extension, including its architecture, the flexible protection signaling, and the secure messaging framework for the communication between the terminal and the tools. Two sample usage scenarios are also provided to illustrate how an MPEG-4 IPMP Extension compliant system works.

Keywords and phrases: digital rights management, multimedia content protection, MPEG4 IPMP Extension, encryption, authentication, interoperable protection.

1. BACKGROUND AND INTRODUCTION

1.1. *Problems in the existing DRM market*

With the advent of digital technologies, many new market opportunities have emerged for content owners, content distributors, and consumer electronics/information technology industries. An essential requirement for developing a thriving marketplace is the protection of copyrighted content in digital form. Digital rights management (DRM) is a technology that has been developed to protect against the illegal distribution of copyrighted digital content such as music, video, or documents. However, there are some problems that remain to be solved in the existing DRM market.

The first problem is the lack of interoperability. Different content providers tend to use different protection mechanisms (hence different DRM systems) to protect and dis-tribute the content. For example, content provider A may prefer to use the Advanced Encryption Standard (AES)[1] for encryption, while content provider B may prefer to use his own proprietary encryption tool. This results in the lack of interoperability as illustrated in Figure 1, where terminal A cannot play back the content distributed by content provider B, and vice versa.

The second problem of the existing DRM market is the lack of renewability. Many existing DRM systems are likely to be broken due to the rapidly growing computer technology. This is one of the serious problems encountered in digital content delivery business. It is therefore desirable to establish a robust and flexible DRM system, where one can easily renew a broken DRM system.

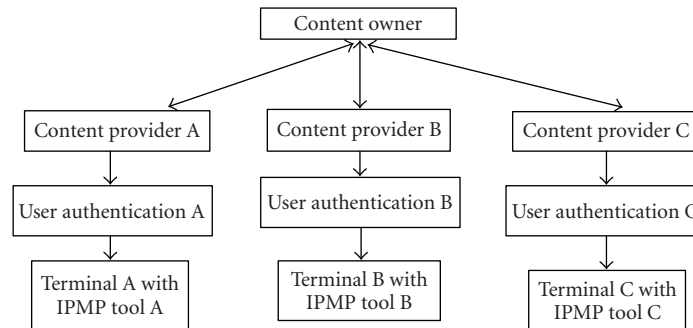[1] NIST US FIPS PUB 197, http://csrc.nist.gov/encryption/aes/.

FIGURE 1: Existing DRM market.

1.2. MPEG-4 IPMP Extension, the answer to the problems

The lack of interoperability problem demands an international standardization effort so that contents can be delivered anytime and to anywhere in the world. Being able to expect different vendors' content to play on a single player is an important matter. Not having to reengineer a given player to work with every other IPMP system is an even more important matter.

With the above considerations, the Motion Picture Experts Group (MPEG), has been pushing for the goal of establishing a DRM standard enabling the functionalities of renewability and interoperability. The MPEG specific term for DRM is intellectual property management and protection, (IPMP). The latest IPMP standard for MPEG-4 system is the MPEG-4 IPMP Extension (IPMPX) [1].

During the development of the IPMP Extension, a real-world scenario that has been discussed intensively, in order to understand more about the scope and the problems that the IPMP Extension should resolve, is the Gobi desert scenario.

Gobi desert scenario. Living in a rather rainy place, Mr. MPEG loves to go to arid places. The Gobi desert is his favorite. Before leaving, imagine that he loads some protected songs on his Panasonic MIEP (MPEG IPMP Extension Player). His wife does the same on her Philips MIEP, but with different songs. When they are in their tent in the middle of the Gobi desert, Mr. MPEG starts listening to his MIEP. He finds a new hit that he feels is great and would like to share it by transferring that song to his wife's MIEP (and, being a rule-abiding guy, he has acquired the rights to do so). Unfortunately, this song has been protected with tools that are new to his wife's MIEP. To make his life harder, there is no Internet connection available in the desert that would allow the required tool to be downloaded to Mrs. MPEG's MIEP. Luckily, being the dictator of MPEG, Mr. MPEG has the power to demand that IPMP Extension support transferring IPMP tools intended for one device to a device of a different make. This would save the trip because otherwise his wife will start asking why he has spent all those years in MPEG if such a simple thing like moving a song from one MIEP to another is not possible and the discussion is likely to degenerate. This demand, however, would make the lives of

the MPEG-4 IPMP committee members miserable, but that is not what Mr. MPEG cares about anyway . . . .

The Gobi desert scenario, explicitly or implicitly, suggests that several factors be considered in the standardization of the MPEG-4 IPMP.

(a) There should be a way to signal to the terminal what IPMP tools are required to consume the contents.

(b) If the required IPMP tools are not available in the terminal, there should be a way to acquire the missing tools from a remote location.

(c) There should be a way to securely transfer the content and the IPMP tools from one device to another.

(d) To ensure interoperability, there should be a way to allow different IPMP tools (potentially from different vendors) to be plugged into the terminal and to interact with each other in a normative manner.

(e) There should be a way to renew the potentially compromised tools.

(f) There should be a way to specify where and to which MPEG-4 content streams the required IPMP tools should be applied and in what order.

(g) There should be a way for the terminal to *securely* communicate with the tools (potentially a plug-in) and to enable tools to communicate *securely* with each other.

(h) There should be a way to convey the IPMP information such as key and rights information to the terminal and to the IPMP tools.

(i) The terminal should comply to the usage rights associated with the user.

(j) Should MPEG-4 IPMP standardize the tools?

(k) Should MPEG-4 IPMP standardize the key management systems?

(l) Should MPEG-4 IPMP standardize the rights management systems?

These issues need to be addressed carefully and in an elegant way to avoid problems experienced in some previous standardization efforts, for example, some technologies chosen by the DVD Forum[2] and the Secure Digital Music Initiative

---

[2]http://www.dvdforum.com/forum.shtml.

(SDMI),[3] an industry forum that intended to develop open technology specifications that protect playing, storing, and distributing of digital music, have been claimed to be hacked. We will show how these considerations have been addressed in MPEG-4 IPMP Extension in the following sections.

1.3. History of the MPEG-4 IPMP Extension

MPEG started its IPMP effort in the development of MPEG-4. The first attempt is often referred to as the "hooks" approach, where normative syntax is defined in MPEG-4 system to allow the bitstream to carry information that informs the terminal which (of possibly multiple) IPMP systems should be used to process the governed objects in compliance with the rules declared by the content provider. The respective IPMP systems themselves were not specified within MPEG-4 [2]. MPEG-4 integrates the hooks tightly with the MPEG-4 systems layer, which makes it possible to build secure MPEG-4 delivery chains in very smart and efficient ways.

This hooks model, however, appears to have many significant problems. For example, IPMP systems can be hooked into the MPEG-4 terminal, but it can only be done on a proprietary basis. Since the protection is normally required to be associated with some elements of the MPEG-4 terminal, and its behavior cannot be independent of other parts of the MPEG-4 terminal, if the IPMP system is not interoperable, an MPEG-4 terminal with IPMP protection would also become non-interoperable.

As a simple example, if the encryption used to protect the video content is different from one IPMP system to another, the consumer electronics (CE) manufacturers would have to build multiple versions of the MPEG-4 terminal to deal with different protection systems used by different content providers. This would significantly increase the cost of building a terminal and, as a result, the consumers would have to bear the high cost. Therefore, the question the MPEG-4 committee faced was whether MPEG can define and standardize an IPMP framework for both the content providers and the CE manufactures to follow so that IPMP systems can become interoperable.

In the year 2000, a new call for proposal (CfP) [3] was issued. Particularly, it aimed to address the interoperability between different products, often for similar services, as developed within the IPMP framework of the MPEG-4 standard. In addition, with convergence becoming a reality, for example, through the deployment of broadband Internet access and the start of new services on mobile channels, interworking between different types of devices and services becomes a more important requirement. The new call requests submission of proposals that would allow interworking between different devices and services designed to play secure digital MPEG-4 content from multiple sources in a simple way, for example, without the need to change the devices.

One issue that particularly needs to be considered when standardizing an IPMP framework in MPEG is the balance between interoperability and security, since these two factors usually contradict each other. Can we standardize every piece of the IPMP system, including a single encryption tool, a single watermarking tool, a single user authentication tool, as well as the key management?

Depending on the scale of the industrial domain and the preference of simplicity or security, one might have different answers to the above question. However, from an international standard (MPEG) point of view, our answer to the above question is no. The first reason is that it will introduce the security issue. For example, sometimes the security of the video watermarking tool depends on the secrecy of the watermarking algorithm, so standardizing a single watermarking tool is not practical. Furthermore, many DRM systems prefer a black-box key management too. Besides the security issue, the second reason is that we have to take care of flexibility as well as renewability. In the current business environment, there are various contents with different importance levels, which are usually protected using different algorithms, such as AES, Data Encryption Standard (DES),[4] and triple DES, e.g., with different security levels. If we would like the same terminal to be able to consume different contents protected with different algorithms, the IPMP framework to be defined has to be flexible. Once the IPMP framework can deal with the flexibility issue, it will be able to support renewability, which is required for IPMP systems for security reason, since an algorithm typically cannot survive many years of attack. After all, MPEG is targeting a large number of industrial domains with different requirements. MPEG4 IPMP should focus on standardizing the most common framework/base for various target applications.

The CfP on the IPMP Extension resulted in numerous submissions from various industries, including many from the authors of this paper. MPEG's systems group has been working with the proponents and started an extension to the MPEG-4 systems standard in the form of an amendment and a new part of MPEG-4 standard. It has reached the Final Draft of International Standard (FDIS) stage in October 2002 [1]. A significant part of the standard was contributed by the authors of this paper.

This paper is organized as follows. Section 2 presents an overview of the architecture of the MPEG-4 IPMP Extension. Sections 3 and 4 detail the core components of the MPEG-4 IPMP Extension. In Section 5, two sample-usage scenarios are presented for an MPEG-4 IPMP Extension compliant system. Section 6 concludes the paper.

2. MPEG-4 IPMP EXTENSION ARCHITECTURE

2.1. Key concepts

It is important to achieve robustness and flexibility in the interoperable framework of a standard. To achieve the robustness, MPEG-4 IPMP Extension provides the tool renewability, which protects against security breakdown. The flexibility

---

[3]http://www.sdmi.org/.

[4]Data Encryption Standard (DES), FIPS PUB 46-3 was reaffirmed in October 1999; http://csrc.nist.gov/publications/fips.

allows the use of various cipher tools as well as decoding tools. The interoperable framework enables the distribution and consumption of content all over the world. MPEG-4 IPMP Extension defines 5 key elements as described below.

(1) IPMP tools

IPMP tools are modules that perform (one or more) IPMP functions such as authentication, decryption, watermarking, etc. A given IPMP tool may coordinate other IPMP tools. Each IPMP tool has a unique IPMP tool ID that identifies a tool in an unambiguous way, at the presentation level or at a universal level.

During the standardization of the IPMP Extension, the MPEG-4 IPMP committee realized that it is not possible to standardize all IPMP tools due to two main reasons. The first is that different content providers have different preferences of the IPMP tools as explained in Section 1.1. The second reason is that there are some tools that are difficult to standardize, for example, it is not possible to standardize a video watermarking tool, as there is no proven robust watermarking algorithm yet. With the above considerations, MPEG-4 IPMP Extension is designed to differ from many prior approaches in that it intelligently provides an *open secure* framework allowing tools from different vendors to cooperate with each other.

(2) IPMP descriptors

This is a part of the MPEG-4 object descriptors (OD) that describe how an object can be accessed and decoded. These IPMP descriptors are used to denote the IPMP tool that was used to protect the object. An independent registration authority (RA) is used so any party can register its own IPMP tool and identify this without collisions.

(3) IPMP elementary stream

IPMP specific data such as key data and rights data are carried by the IPMP elementary stream (ES). All MPEG objects are represented by ES, which can reference each other. These special ES can be used to convey IPMP specific data. Their syntax and semantics are further specified in MPEG-4 IPMP Extension [1].

(4) IPMP tool list

IPMP tool list carries the information of the tools required by the terminal to consume the content. It is carried in the initial object descriptor (IOD) of the MPEG-4 system stream. This mechanism enables the terminal to select, manage the tools, or retrieve them when they are missing, and so forth [4].

(5) Secure messaging framework

The MPEG-4 IPMP Extension framework did not choose the approach of defining functional interfaces. Instead, it is based on secure message communication [1]. This is one of the most important concepts in MPEG-4 IPMP Extension. Interaction between the terminal and the IPMP tools are realized through the messages via a conceptual entity called message router (MR). Syntax and semantics of the messages are clearly defined to facilitate full interoperability. Mutual authentication and secure messages are also introduced to achieve a secure framework. Note that the normal functional interfaces are unlikely to cover various kinds of interfaces for different algorithms, even for the same encryption function. Furthermore, the normal functional interfaces are highly dependent on the operating system and the implementation.

The message-based architecture has three advantages over functional interface-based architectures. The first is that security can be more easily maintained, as messages are easier to protect in an open framework than the parameters in a function parameter list. The second is that the only entities that need to be concerned with a given message's definition are those that need to generate or act upon a given message; so additional functionality can be created and supported simply through the addition of the required messages. The third is that full interoperability with IPMP tools can be easily achieved by registering the messaging API to a RA and carrying the registered API ID in the IPMP_ToolAPI_Config information in the IPMP descriptor, or by defining a single messaging API by a third-party forum which adopts MPEG-4 IPMP Extension. Note that MPEG is not taking the role of defining a single messaging API since MPEG is targeting a large number of industrial domains. Individual industrial domains should take MPEG-4 IPMP Extension as a base, and fill in the gap in order to make IPMP Extension truly interoperable.

Note that in the hooks approach [2], MPEG-4 IPMP defines how an object is treated and how the IPMP specific data are carried. In other words, (2) and (3) discussed above are included in the hooks approach. In the IPMP Extension, (4) and (5) are added while (2) and (3) are further improved, and the concept of IPMP system in IPMP hooks is changed to that of IPMP tool as discussed in (1). IPMP Extension enhances the original hooks approach so that tool renewability and flexibility can be achieved.

Considering the diverse applications (e.g., real-time communications, Internet streaming, surveillance, broadband, wireless, studio, DVD, set-top box, etc.) that MPEG-4 intends to address [5], it is very difficult to have a complete "one-fits-all" solution. For example, as discussed above, it would be very difficult to standardize tools in MPEG, a standardization body whose main mission is to standardize core technologies, rather than metadata or making business decision. Instead, MPEG-4 chose to standardize a flexible architecture that would allow individual industries to extend the framework and further define their own complete standards to achieve full interoperability, based on the requirements of the individual industry and business consideration. For example, key management and user registration/authentication are not defined in MPEG-4 IPMP Extension. Their implementations are up to the IPMP tools on top of MPEG-4 IPMP Extension. This enables using different IPMP tools for different applications while providing a common framework to facilitate the support of full interoperability.
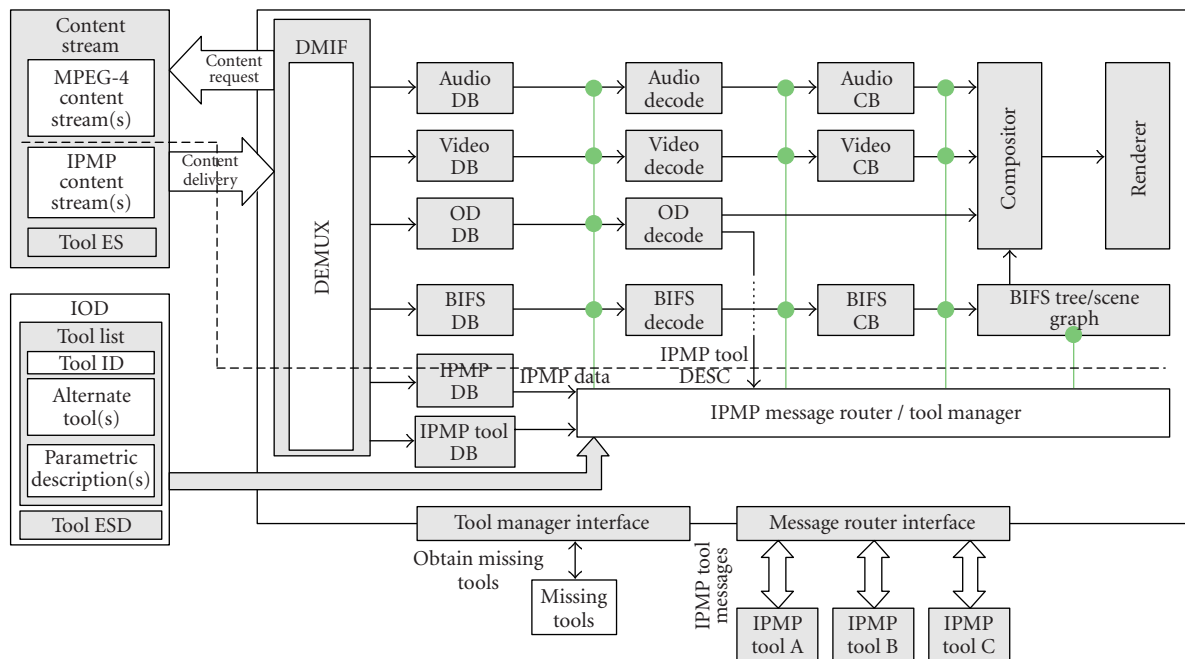
FIGURE 2: The MPEG-4 IPMP terminal architecture.

2.2. Architecture

Figure 2 shows the terminal architecture under the MPEG-4 IPMP Extension framework. The original MPEG-4 system without IPMP protection is shown in the upper half of the diagram (above the dotted line). The incoming MPEG-4 content stream is demultiplexed in the delivery multimedia integration framework (DMIF). Audio, video, OD, and binary format for scenes (BIFS) bitstreams are supplied to the decoding buffers (DB) and then decoded. The decoded audio and video data are fed to the audio composition buffer (CB) and the video CB, respectively, and then are composed in the compositor together with the decoded ODs and the decoded BIFS tree or scene graph.

The lower half of the figure (below the dotted line) shows the modules provided by the IPMP Extension. The tool list is included in the IOD of the MPEG-4 system stream to identify the IPMP tools required to consume the protected content. IPMP stream arrives as an ES multiplexed in the MPEG-4 system stream. Note that the tool list and the IPMP stream are constructed during the content authoring process (see Section 5.1.1 for an example). The tool manager (a conceptual entity) manages IPMP tools within the terminal (e.g., downloading a missing tool from a remote location) while MR routes messages among the terminal and the IPMP tools using a secure messaging framework (to be introduced in Section 4) to ensure that different IPMP tools from different vendors can work together. IPMP tools can act on several control points, which are positions along the dataflow where the IPMP tool functions by taking over the protected content bitstream, processing it, and returning it to the control point for subsequent processing of the content by the MPEG-4 terminal. The supported control points are dictated by the

gray circles in the architecture diagram. For example, an encrypted MPEG-4 video stream needs to be decrypted by an IPMP tool (decrypter) at the control point right before the video decoder, and a watermark reader may need to be applied to the watermarked audio stream at the control point right after the audio decoder. If necessary, an IPMP tool can be applied to the control points right before the compositor to control the rendering process. Details about how to signal the protection scope (which objects or ESs) and the control points of the IPMP tools when authoring the MPEG-4 content stream are presented in Section 3.2.

2.3. Advantages of the IPMP extension architecture

The IPMP Extension architecture achieves several important functionalities.

Interoperability

MPEG-4 IPMP Extension standardizes the IPMP messages and the process of message routing. By using a common set of IPMP messages, together with *industry defined* (not MPEG-4 IPMP defined) messaging API and messages extension, different IPMP tools can be easily plugged into the terminal and interact with each other.

Renewability

Through the usage of the tool list and IPMP descriptor, one can easily renew a tool for better IPMP protection by, for example, indicating to the terminal that a new tool is needed, carrying the new tool in the tool ES in the content stream, or downloading the new tool from somewhere. Note that tool downloading is not mandatory in IPMP. IPMP provides the architecture to facilitate tool downloading.
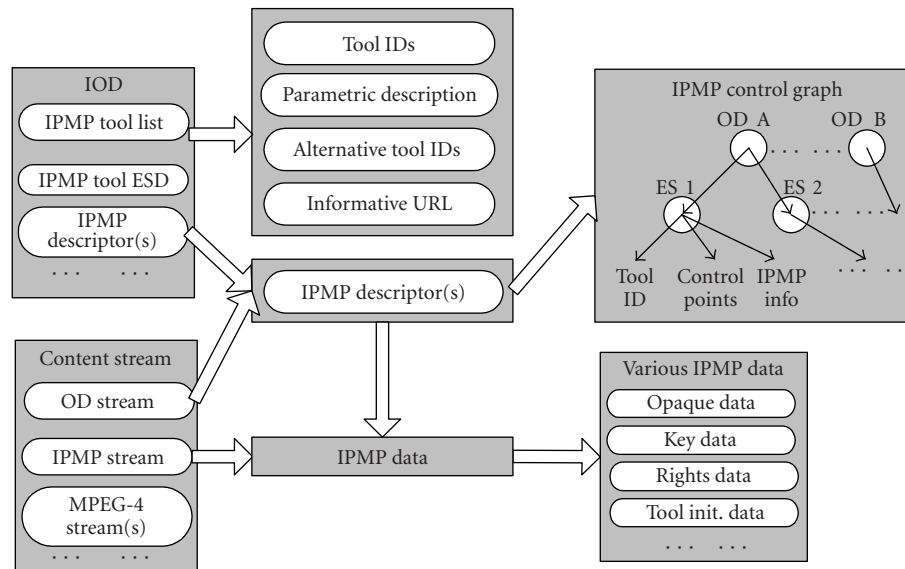
FIGURE 3: Structure of an MPEG-4 system content protected by IPMP Extension.

Flexibility

MPEG-4 IPMP Extension does not standardize the tools. With the support of independent RAs, the ability to carry tools inside the content stream, and the terminal's potential capability to download required IPMP tools from a remote location, one can choose whatever algorithms or tools to perform decryption, watermarking, user authentication, or integrity checking.

Dynamic operation

Various IPMP tools protection can be signaled in the content with the help of IPMP descriptor, control point, and sequence code (see definition in Section 3.2.1). Different tools can operate at the same or different control points, acting on the same or different streams.

Secure tools

Terminal and tools can choose to perform mutual authentication using the IPMP authentication messages (see discussion in Section 4.2.5) to achieve a secure communication framework.

3. FLEXIBLE PROTECTION SIGNALING

Figure 3 illustrates the structure of an MPEG-4 system content protected by IPMP Extension. The information contained in the IOD and the content stream is shown and the relation between them is indicated. More details about each entity in Figure 3 will be described in the following.

3.1. Required IPMP tools and carriage of IPMP tools

3.1.1. IPMP tool list

The idea of IPMP tool list [4] is an improvement over the IPMP hooks. MPEG-4 IPMP Extension defines a syntactic description language (SDL) [6] descriptor IPMP_ToolList-

Descriptor in IOD which supports the indication of independent or alternative IPMP tools required to consume the protected content. IOD is chosen to carry the IPMP tool list since IOD arrives ahead of OD, BIFS, and other ESs, hence allows the IPMP tool manager to retrieve and make sure every IPMP tool is present.

For each tool in the IPMP tool list, the following information is provided:

(i) IPMP tool identifier: a given IPMP tool is identified to other entities via its IPMP tool identifier;
(ii) possible alternatives to a given tool;
(iii) optional parametric description of the tool (i.e., information that enables a terminal to choose a specific tool implementation);
(iv) optional informative URL.

The above structure of the IPMP tool list provides the terminal sufficient information to retrieve a tool that is required to consume the protected content. It also provides a flexible way to identify an IPMP tool via its alternatives or parametric description [1].

3.1.2. IPMP tool ESD

The IPMP tools required to consume the protected content may have already been in the terminal, or may be downloadable from a remote location. One or more binary representations of IPMP tools may also be carried directly or by reference in an MPEG presentation. MPEG-4 IPMP Extension defines a new ES with stream type "IPMPToolStream" for carrying binary IPMP tools within an MPEG-4 system stream.

One implementation of a given tool is carried as the payload of one IPMP tool ES whose representation format, packaging information, and IPMP tool ID are specified in DecoderConfigDescriptor in the associated elementary stream descriptor (ESD).

The IPMP tool ES is referenced through the IOD, as illustrated in Figure 3. The IPMP tool manager serves as a decoder for the IPMP tool ESs. IPMP tools carried within the IPMP tool ES can be installed, used and retained at the discretion of the terminal implementation. They are referenced via their IPMP tool IDs just like any other IPMP tool.

3.2. Signaling of various IPMP tools protection scope

It is necessary to signal in the MPEG content stream which objects or ESs a particular IPMP tool should be used to protect, and where in the dataflow of the MPEG-4 terminal the tool should be applied. The signaling of the protection scope and its control point inherits from the IPMP hooks [2] by using IPMP descriptors and IPMP descriptor pointers. However, both IPMP descriptor and IPMP descriptor pointer have been improved to allow a more flexible indication and to provide more functionality.

3.2.1. IPMP descriptor

The IPMP_Descriptor carries IPMP information for one or more IPMP tool instances. It may also contain optional instantiation information for one or more IPMP tool instances. IPMP_Descriptors are conveyed and updated in either IODs, ODs, or OD streams.

Each IPMP_Descriptor has an IPMP_ToolID, which identifies the required IPMP tool for protection. The control point of the IPMP tool's protection is signaled by another element in IPMP_Descriptor: controlPointCode, which specifies where the IPMP tool resides (see control points illustrated in Figure 2).

Sequence code [7] is another element in IPMP_Descriptor that is used to signal the sequencing priority of the IPMP tool instances at the given control point. In the case that multiple tools are governing the same control point on a given stream, the tool with the highest sequence code will process the data first for that control point on that stream.

3.2.2. Using IPMP descriptor to signal protection at different control points

The IPMP_DescriptorPointer appears in the ipmpDescPtr section of an OD or ESD structure. Different presence locations signal different protection scopes. The presence of this descriptor pointer in an OD indicates that *all* streams referred to by embedded ES_Descriptors are subject to protection and management by the IPMP tool specified in the referenced IPMP_Descriptor. The presence of this descriptor pointer in an ES_Descriptor indicates that *only* the stream associated with this descriptor is subject to protection and management by the IPMP tool specified in the referenced IPMP_Descriptor.

IPMP_DescriptorPointer also has an IPMP_ES_ID that is the ID of an IPMP stream that may carry messages intended to the tool specified in the referenced IPMP_Descriptor. In case more than one IPMP stream is needed to feed the IPMP tool, several IPMP_DescriptorPointers can be given with the same IPMP_DescriptorID and different IPMP_ES_IDs.

By utilizing the IPMP_Descriptor and IPMP_DescriptorPointers, the terminal can build an abstract IPMP control
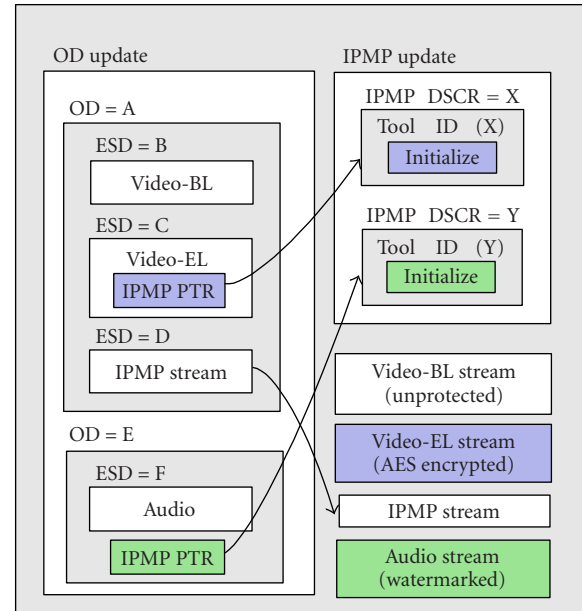

FIGURE 4: A sample content structure.

graph (see Figure 3), which bears a tree-like hierarchy. One example is shown in Figure 4, where an ES Video-EL stream is associated with the ESD = C under OD A. OD A contains an IPMP descriptor pointer that points to an IPMP descriptor (IPMP DSCR = X) which carries tool ID of the IPMP tool required to consume the VIDEO_EL stream, information about where the IPMP tool should be applied (i.e., control points), and other IPMP information. Different IPMP tools can be specified to protect different objects or different ESs under that object, at different control points, or at the same control point but bearing different sequence codes.

3.3. Delivery of IPMP data to the terminal and/or IPMP tools

IPMP data is the information directed to a given IPMP tool or terminal to enable, assist, or facilitate its operation. It is sometimes referred to as IPMP information. IPMP data includes but is not limited to key, usage rights, tool initialization, and mutual authentication information [8].

3.3.1. Places to carry IPMP data

IPMP data can come from various sources. When it is carried in the content, it can be contained in IPMP_Message class in an IPMP stream or IPMP_Descriptor [1]. IPMP_Message is the data class defined to carry IPMP data in the IPMP stream, which includes the identification of the recipient of this IPMP_Message as well as a place holder for IPMP data to be carried inside.

IPMP data can also be generated by an IPMP tool or IPMP terminal and delivered to other IPMP tools or the IPMP terminal as a payload of IPMP_MessageFromTool (see definition in Section 4.2.1).

3.3.2. Delivery of IPMP data to IPMP tools

IPMP information is routed using *normative* addressing methods, as discussed in Section 4.2. The addressee of a specific message is implicit either by bitstream context or by process context. In the MPEG-4 bitstream context, the addressee is the IPMP tool whose identity is indicated in the IPMP message or IPMP descriptor header. Information is delivered at a specific time, specified in the bitstream or implicit by process.

IPMP data carried in the IPMP_Descriptor is delivered to the IPMP tool declared in the descriptor. The IPMP data is sent as a payload of the message IPMP_DescriptorFromBitstream (see definition in Section 4.2.1). IPMP data carried in IPMP_Message class of IPMP stream is delivered to the IPMP tool declared in the IPMP_Descriptor whose IPMP_DescriptorID is indicated in the same IPMP_Message class. The IPMP data is sent as a payload of the message IPMP_MessageFromBitstream (see definition in Section 4.2.1).

Physical routing of information and context resolution are handled by the MR. The MR abstracts all platform-dependent routing and delivery issues from the IPMP tools.

4. SECURE MESSAGING FRAMEWORK

MPEG-4 IPMP Extension defines the following components of the IPMP tool interaction framework: interaction (or communication) between the terminal and the IPMP tool(s), realized via "messaging" between the terminal and the IPMP tools; the messages (syntax and semantics); and the process of message routing. As discussed in Section 2, this messaging framework allows different IPMP tools, potentially from different vendors, to be easily plugged into the terminal and to interoperate with each other and with the terminal in a secure way. This is a critical step toward supporting interoperability in MPEG-4 IPMP.

All IPMP tool interactions take place via the terminal. IPMP tools do not communicate directly with each other within the scope of the standard.

4.1. Flexible messaging infrastructure

All IPMP tool messages are routed through the terminal. To represent this function, an entity called the MR is defined in the architecture. The MR connects and communicates with supported IPMP tool(s). It thus abstracts the physical interface of one IPMP tool from any other IPMP tool that wishes to communicate with it. The interface between the MR and the tools is *nonnormative* and is not defined in the specification. Only messages derived from an expandable base message class called IPMP_ToolMessageBase [1] may cross the interface.

Message routing is assumed to be instantaneous. In case of an MR error, an appropriate error status is returned by the MR. In all other cases, the MR is required to route, without a change in semantic meaning, information and responses as received.

4.2. Messages defined within MPEG-4 IPMP Extension

IPMP_ToolMessageBase is the expandable base class for all messages that may across the messaging interface within MPEG-4 IPMP Extension. It specifies the context ID (identifier of the logical instance of a tool assigned by the terminal) of the originator of the message, and the context ID of the intended recipient of the message.

4.2.1. IPMP data delivery messages

There are currently three defined IPMP data delivery messages [1], that is, IPMP_MessageFromBitstream, IPMP_DescriptorFromBitstream, and IPMP_MessageFromTool. Message IPMP_MessageFromBitstream is used to deliver IPMP_Messages received in the content to the IPMP tool context specified in the IPMP_Message. If an IPMP *access unit* delivered in the IPMP ES contains more than one IPMP_Message for a specific IPMP tool, all IPMP_Messages for that tool will be included in a single IPMP_MessageFromBitstream message. Note that Access Unit is one individually accessible portion of data within an ES. An access unit is the smallest data entity to which timing information can be attributed. Message IPMP_DescriptorFromBitstream is used to deliver an IPMP_Descriptor received in the bitstream to the IPMP tool specified in the IPMP_Descriptor.

Message IPMP_MessageFromTool is used to deliver any IPMP data from tool to tool. These IPMP data can be categorized into instantiation and notification messages, event notification messages, IPMP processing messages, authentication messages, user interaction messages, consumption messages, and inter-device messages.

4.2.2. Instantiation and notification messages

These messages are used to instantiate and destroy logical instances of new tools, to inform newly instantiated tools of existing tools, and to notify existing tools of a new instantiation. Although they are primarily designed to be used by tools to request logical instances of other tools, these messages may also be used in the content stream when upstream capabilities exist, for example, for mutual authentication between the server and the terminal.

4.2.3. Event notification messages

These messages provide the IPMP tools with the ability to request and get notified of events including connection, disconnection, and watermark detection.

4.2.4. IPMP processing

These messages are defined to be used in the IPMP process. Although the exact functioning of the various IPMP tools is not specified, these messages support the *interoperable* use of common types of IPMP tools such as encryption/decryption, audio and video watermarking, as well as rights management and governance. For example, the IPMP_SelectiveDecryptionInit message defined in Annex A

of the MPEG-4 IPMP Extension [1, 9, 10] allows a terminal to configure a selective decryption tool (see, e.g., the ones proposed in [11, 12]). It tells how the bitstream is encrypted, whether all bits are encrypted, or only portions of it, what portions of the received bitstream are encrypted [11] or shuffled [12], and therefore need to be decrypted or deshuffled, and so forth. The IPMP_KeyData message allows carriage of a key, including timing information in order to synchronize the key with the media stream. These messages may be directly carried in the bitstream in the IPMP_Message and/or the IPMP_Descriptor messages or may be wrapped in the IPMP_MessageFromBitstream class or IPMP_DescriptorFromBitstream messages for passing between tools or between tools and the terminal.

4.2.5. Authentication messages

At any point in IPMP information or content processing, IPMP tools may be required to communicate with one another or with the terminal. The degree of security required for such communication is determined by a number of variables including information that may be included by the content provider in the content and conditions of trust established between tool providers a priori and out of band. It is generally the case that a given ES is protected by multiple tools but that certain types of tools are complex (e.g., rights management tools) and others are utilities (e.g., decryption engines). Complex tools may control the instantiation of other tools or make decisions about content use in response to usage queries from the terminal. Mutual authentication may occur between any pair of tools but the level of security required for this communication will in part be dictated by data contained in the bitstream in an opaque manner. The mechanism for making the determination of this security level is nonnormative.

Mutual authentication is executed as follows.

(1) The tool that initiates mutual authentication with another tool determines the conditions of trust to be achieved by such authentication, that is, the initiating tool determines whether it needs only integrity-protected communication or fully secure, authenticated communication. This level may or may not be dictated by IPMP information in the content.

(2) The communicating tools then engage in a message exchange to determine which authentication protocol will be used. In some cases, this protocol may have been determined by an a priori out-of-band negotiation between the tool providers in their security audits of one another. The authentication messages are used to request a mutual authentication, or are generated by and exchanged between IPMP tools and IPMP tools and a terminal for the purpose of mutual authentication.

4.2.6. User-interaction messages

These messages allow information to be exchanged between the user and an entity requiring information from the user.

4.2.7. Consumption permission

The IPMP_CanProcess message enables the notification of the terminal, by IPMP tools, as to the tools ability to begin or discontinue processing content.

4.2.8. Inter-device messages

MPEG-4 IPMP Extension has also defined a set of inter-device messages in [1, Annex D]. These messages support the transfer of the content and IPMP tools. Transfer of the content and tools can be made secure by putting them into secure message payload, using any established mechanisms. Section 5.2 makes use of these messages to provide a solution to the Gobi desert scenario.

5. TWO SAMPLE USAGE SCENARIOS

We illustrate two sample usage scenarios in this section where the second one is the usage scenario for the Gobi desert scenario we discussed in Section 1.

The first sample usage scenario illustrates a use case whereby an MPEG-4 system stream consists of one video object and one audio object. The video object is further composed of two ESs, one is video stream base layer (BL), while the other is video stream enhancement layer (EL). It is protected by MPEG-4 IPMP Extension.

5.1. A simple MPEG-4 IPMP Extension protected MPEG-4 content

5.1.1. Content authoring

At the content creation side, the content author creates a simple MPEG-4 system stream, which mainly consists of one single audio object with one audio ES under it and one single video object with two video ESs (BL and EL) under it.

In order to protect the content, the content author uses AES [13] encryption tool to encrypt the video EL since it is of a higher value. The video BL remains unprotected since it is not of a high commercial value. The author also embeds (using watermark encoding) some copyright information bits into the audio stream.

Suppose that the content author is aware that there are an IPMP tool X with tool_id AAA that is capable of doing AES decryption and an IPMP tool Y with tool_id BBB that is able to detect the watermark from the audio ES. The content author hence constructs the IPMP tool list including the above-mentioned two tool_ids to indicate to any terminal receiving the MPEG-4 content that these two tools are needed to play the content. The tool list descriptor is put under IOD. If necessary, the author can also put IPMP tool Y, binaries compiled for the desired platforms, as a tool ES referenced in IOD, in case the terminal does not have tool Y.

The content author constructs the abstract IPMP control graph (described in Section 3.2.2) using IPMP_Descriptor and IPMP_DescriptorPointer to indicate to the terminal that tool X needs to be used for video EL stream and that tool X needs to sit at the control point of "before_decoder." The control graph also indicates that tool Y needs to be used for audio ES and that tool Y needs to sit at the control point of
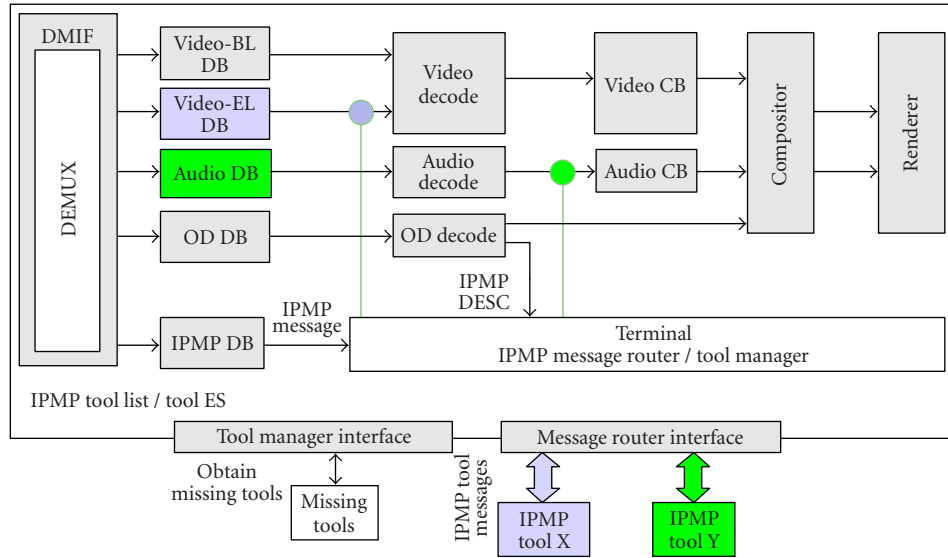
FIGURE 5: A sample terminal architecture.

5.1.2. MPEG-4 IPMP Extension terminal behavior

"after_decoder." The IPMP control graph can be built by embedding IPMP descriptor pointers into their respective ESD or OD. The control point information, sequencing code, as well as any opaque data which may contain the tool initialization information are carried in each tool's specific IPMP descriptor which is sent through OD stream.

The AES encryption uses a time-variant key stream to encrypt the above-mentioned video EL stream. Hence the content author constructs the IPMP stream which is a concatenation of IPMP_Message class, with each IPMP_Message specifying the destination (i.e., IPMP tool X), and each IPMP_Message body containing IPMP_KeyData which carries the time-variant key. The constructed IPMP stream is also multiplexed with other ES under the video object (see OD A in Figure 4). The content structure is shown in Figure 4. IOD and BIFS are omitted for brevity.

5.1.2.   *MPEG-4 IPMP Extension terminal behavior*

The simplified architecture of MPEG-4 IPMP Extension terminal consisting of the two tools to handle the above authored content is demonstrated in Figure 5.

The dataflow and how IPMP tools protect the content are based on Sections 3 and 4. The terminal, upon receiving the above-constructed IPMP protected MPEG-4 stream, retrieves the IPMP tool list from IOD. According to the two tool_ids mentioned in the IPMP tool list, the tool manager checks the presence of the two tools inside the terminal. If not present, the tool manager may retrieve them from a remote location which is also indicated in the tool list, it may attempt to get the missing tool from neighboring devices, or may retrieve the tool from the content (if the tool is carried in the content as a tool ES).

The terminal then checks the IPMP control graph by retrieving IPMP descriptor pointers from the OD and/or ESD. The IPMP descriptors pointed by the two pointers are up-

dated through OD stream. It now has the information on where and how tool X and tool Y should be used.

Tool X is instantiated at the before_decoder control point (between Video-EL DB and the video decoder). Tool Y is instantiated at the control point that is after the audio decoder. Both tools need to do a mutual authentication with the terminal using the mutual authentication messages to ensure both tools are trusted by the terminal. The mutual authentication could result in a secure communication channel between IPMP tools and the terminal.

The IPMP descriptor containing the control point, sequence code, and other IPMP data is sent to the tool indicated in the IPMP descriptor through the IPMP_DescriptorFromBitstream message. The IPMP data embedded in the IPMP descriptor may include the initialization information for that particular tool, for example, IPMP_AudioWatermarkingInit [1]. The IPMP tool receives this information and configures itself.

At the control point of Video-EL decryption, the terminal routes demultiplexed Video-EL bitstream to the IPMP tool X running at that control point.

The IPMP stream is received by the terminal. According to the destination address (IPMP descriptor ID) contained within each IPMP_Message($\cdot$), the message is routed to the specific tool at the time indicated by the timing information associated with the access unit which carries the IPMP_Message($\cdot$).

The delivery is done using IPMP_MessageFromBitstream message. For the IPMP tool X (AES decryption tool), the message contains the time-variant key in the form of IPMP_KeyData, which is used by tool X to do its decryption job.

After receiving and decrypting the Video-EL access units, the IPMP tool X returns the decrypted-video access units to the terminal through the *nonnormative* messaging interface.

At the control point of the audio watermark retrieval, the terminal routes every decoded audio packet to the IPMP tool Y. Tool Y retrieves the watermark from the received audio packets, and the watermark retrieval result is notified to the terminal in the form of IPMP_SendAudioWatermark message [1]. Tool Y may also verify the copyright information bits in the audio stream and, if necessary, tool Y can control the rendering process by sending the IPMP_CanProcess message to the terminal.

5.2. A note on the Gobi desert scenario

In the Gobi desert scenario, it is assumed that two different devices (owned by Alice and Bob) want to share content and that they can communicate with one another via IR, firewire, and so forth. Alice's device supports IPMP-A tool, Bob's supports IPMP-B tool. The following steps show how this is accomplished within the MPEG-4 IPMP Extension framework.

(1) Bob wants to listen to the content that is packaged for IPMP-A.

(2) He connects his device to Alice's.

(3) He locates the content that he wants and requests a download through the MPEG-4 IPMPX defined interdevice messages.

(4) Alice and Bob's devices do a mutual authentication using IPMP Extension's interdevice messages, and establish a secure authentication channel (SAC).

(5) Alice's device transfers the content to Bob's device using the secure messages over the SAC between the two devices.

(6) By checking the IPMP tool list in the requested content, Bob's device determines that IPMP-A tool is required and that IPMP-A tool is not available in the terminal nor is it conveyed in the IPMP tool ES in the content stream.

(7) Bob's device connects to Alice's device to request the missing IPMP-A tool.

(8) Again, mutual authentication is done between Alice and Bob's devices and a SAC is established.

(9) The IPMP-A tool is securely transferred to Bob's device using IPMPX's interdevice tool transfer messages.

(10) Bob can now play the content locally by using the IPMP-A tool.

(11) Note that the trust relationship between the two devices should have been established between the device manufacturers. If the two devices do not trust each other, this copy procedure cannot occur.

6. CONCLUSION

This paper introduces MPEG-4 IPMP Extension, the breakthrough technology standardized by MPEG for interoperable DRM. MPEG-4 IPMP Extension offers flexibility, robustness, and interoperability, promotes secure content delivery around the globe. MPEG-4 IPMP Extension can be used in combination with proprietary tools, which enables the implementation of various degrees of security for different business models while maintaining the interoperability. Some implementation issues, such as messaging interfaces,

RA, and profiling for different industrial domains, are considered out of the scope of MPEG and are left unspecified. They are left for further specification by the industrial body for a specific application.

MPEG-4 IPMP Extension has been finalized, and the industry is beginning to accept it. MPEG open security for embedded systems (MOSES),[5] a consortium of more than 7 worldwide companies, has just launched a *music-4-you* service based on MPEG-4 IPMP Extension for secure music distribution. Internet Streaming Media Alliance (ISMA)[6] has adopted MPEG-4 IPMP Extension's protection signaling method in its ISMACryp specification. The MPEG-4 IPMP Extension framework has also been successfully mapped to MPEG-2 system, resulting in MPEG-2 IPMP [14, 15], which has drawn substantial interest from the broadcasting industry as well as broadband applications.

REFERENCES

[1] MPEG-4, "Information technology—coding of audio-visual objects—Part 13: Intellectual property management and protection (IPMP) extension," ISO/IEC JTC1/SC29/WG11, N5284, Shanghai, China, October 2002.

[2] MPEG-4, "MPEG-4 intellectual property management & protection (IPMP) overview & applications," ISO/IEC JTC1/SC29/WG11, N2614, Rome, Italy, December 1998.

[3] MPEG, "Call for proposals for IPMP solutions," ISO/IEC JTC1/SC29/WG11, N3543, Beijing, China, July, 2000.

[4] M. Ji and S. M. Shen, "Refined tool list syntax and tool download ability," MPEG, ISO/IEC JTC1/SC29/WG11, M7530, Sydney, Australia, July 2001.

[5] MPEG-4, "MPEG-4 applications," ISO/IEC JTC1/SC29/WG11, N2195, Tokyo, Japan, March 1998.

[6] MPEG, "Information technology—Generic coding of moving pictures and associated audio—Part 1: Systems," ISO/IEC JTC1/SC29/WG11, International Standard ISO/IEC 14496-1, 2001.

[7] M. Ji and S. M. Shen, "Proposal on improvement of the current MPEG4 IPMP Extension CD," *ISO/IEC JTC1/ SC29/WG11 MPEG,* M7569, Santa Clara, Cailf, USA, October 2001.

[8] M. Ji and S. M. Shen, "IPMP Data Base Class and Video Watermarking Message," *ISO/IEC JTC1/ SC29/WG11 MPEG,* M8342.

[9] W. Zeng, J. Wen, and M. Severa, "Editorial changes and extension of Annex C on selective decryption configuration message," *ISO/IEC JTC 1/SC 29/WG 11/IPMP* M7989, Jeju, Korea, March 2002.

[10] J. Wen, W. Zeng, D. Kosiba, and M. Severa, "A format-compliant configurable encryption framework for access control of multimedia," *ISO/IEC JTC 1/SC 29/WG 11/IPMP* M7213, Paris, France, June 2001.

---

[5] http://www.crl.co.uk/projects/moses/.

[6] http://www.isma.tv.

[11] J. Wen, M. Severa, W. Zeng, M. H. Luttrell, and W. Jin, "A format-compliant configurable encryption framework for access control of video," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 12, no. 6, pp. 545–557, 2002.

[12] W. Zeng, J. Wen, and M. Severa, "Fast self-synchronous content scrambling by spatially shuffling codewords of compressed bitstreams," in *Proc. IEEE International Conference on Image Processing (ICIP '02)*, vol. 3, pp. 169–172, Rochester, NY, USA, September 2002.

[13] Advanced Encryption Standard, NIST US FIPS PUB 197, http://csrc.nist.gov/encryption/aes/.

[14] MPEG, "Study text of ISO/IEC 13818-11/FCD (MPEG-2 IPMP)," ISO/IEC JTC1/SC29/WG11, N5469, Awaji, Japan, December 2002.

[15] MPEG, "Study text of ISO/IEC 13818-1:2000/FPDAM2 (MPEG-2 IPMP amendment)," ISO/IEC JTC1/SC29/WG11, N5465, Awaji, Japan, December, 2002.

Ming Ji graduated from the Department of Electrical and Computer Engineering, National University of Singapore, in 1999; he then worked as Research & Development (R&D) Engineer for Singapore Technologies for one year, and has been working as a Senior R&D Engineer at Panasonic Singapore Laboratories for 4 years. He has been involved in video coding, video streaming, watermarking, and digital rights management (DRM) related development and research activities. During 2000–2003, he joined the MPEG IPMP standardization and made important contributions. He was the Editor for MPEG-2 IPMP specification and one of the coeditors for MPEG-4 IPMP specification. He is also participating in other DRM standardizations.

S. M. Shen worked as a Senior Staff Engineer and now she is working as the Manager of Panasonic Singapore Laboratories in Singapore. She has been working in MPEG-1/-2/-4 standardization and related product developments for 13 years, mainly in video coding. During 2000–2003, she joined in MPEG IPMP standardization and made important contributions with her team. She also led a team to work in DTV, content distribution and management, as well as audio product development. She received her Bachelor degree in electrical engineering and her Master degree in adaptive signal processing from North-West Telecommunications Engineering Institute, in Xi'an (now Xidian University) in 1984 and 1986, respectively. She worked in Electrical Engineering Laboratories at the same University for two years before she went to Japan where she worked for 3 years in the area of medical signal processing.

Wenjun Zeng received his B.E., M.S., and Ph.D. degrees from Tsinghua University, China, in 1990, the University of Notre Dame in 1993, and Princeton University in 1997, respectively, all in electrical engineering. He is currently an Associate Professor at the Computer Science Department, University of Missouri-Columbia. He worked at Matsushita Information Technology Lab, Panasonic Technologies Inc., Princeton, in 1995, and at Multimedia Communication Lab, Bell Laboratories, Murray Hill, NJ, in 1996. From 1997 to 2000, he was with Sharp Labs of America, Camas, Washington. He was with PacketVideo Corporation, San Diego, from December 2000 to August 2003, where he was leading Research & Development projects on wireless multimedia streaming, encoder quality optimization, and digital rights management. His current research interests include multimedia communications and networking, content and network security, and wireless multimedia. He has been an active contributor to the JPEG 2000 and the MPEG4 IPMP Extension standards, where four of his proposals have been adopted. Dr. Zeng has served as a Special Session and Panel Session Organizer for several IEEE international conferences. He was the Lead Guest Editor of IEEE Transactions on Multimedia's Special Issue on Streaming Media published in April 2004. He is an Associate Editor of the IEEE Transactions on Multimedia, and is the Technical Program Cochair of the Multimedia Communications and Home Networking Symposium and 2005 IEEE International Conference on Communications.

Taka Senoh received his M.S. degree in communication engineering from Osaka University, Japan, in 1975. In the same year, he joined Matsushita Electric Industrial Co., Ltd. Since then, he has been working on digital audio and video coding technology development. He was a visiting scientist at MIT Media Lab, US, from 1988 till 1990. He is a member of IEEE, IEICE (Institute of Electronics, Information & Communication Engineers of Japan), and ITE (Institute of Image Information & Television Engineers of Japan).

Takafumi Ueno received his M.S. degree in electrical engineering from Nagoya University in 1974. In 1974, he joined Matsushita Electric Industrial Co., Ltd. In 1984, he received his Ph.D. degree on information electronics from Nagoya University of Japan. In Matsushita, he has been engaged in digital technologies on computer software, digital video/audio, optical disc, and the international standardization activity. He is now engaged in the development of digital rights management and content distribution technology including the MPEG standardization and its implementation to home and mobile environments at AVC Development Center under Panasonic AVC Networks Company of Matsushita Electric Industrial Co., Ltd. He is a member of the IEEE (Institute of Electrical and Electronics Engineers).

Terumasa Aoki is a Lecturer at the Research Center for Advanced Science and Technology, the University of Tokyo. He received his B.S., M.E., and Ph.D. degree in information and communication from the University of Tokyo, Japan, in 1993, 1995, and 1998, respectively. His current research interests are in the fields of Terabit IP router, access control of Gigabit LAN/WAN, next-generation video conferencing system, high-efficient image coding, and management of digital content copyrights. He has received various academic excellent awards such as the 2001 IPSJ Yamashita Award, the FEEICP Inose Award for 1994, and another 4 awards.

Yasuda Hiroshi received his B.E., M.E., and Dr.E. degrees from the University of Tokyo, Japan, in 1967, 1969, and 1972, respectively. Then, he joined the Electrical Communication Laboratories of NTT in 1972. After serving twenty-five years from 1972 to 1997), with the last position being the Vice President and Director of NTT Information and Communication Systems Laboratories at Yokosuka, he left NTT and joined The University of Tokyo. He is now the Director of The Center for Collaborative Research (CCR), and the Professor in Research Center for Advanced Science & Technology. His study area is in applied information technology. He has been involved in works on video coding, image processing, tele-presence, B-ISDN network and services, and internet and computer communication applications. Now he has started researches on digital rights management (DRM) and "Kansei" (more human) communication. In international standardizing area, he served as the Chairman of ISO/IEC JTC1/SC29 (JPEG/MPEG Standardization) from 1991 to 1999. He had also served as the President of Digital Audio Video Council (DAVIC) from September 1996 to September 1998. He received 1987 Takayanagi Award, the 1995 Achievement Award of EICEJ, the 1995–1996 EMMY from The National Academy of Television Arts and Science, and also 2000 Charles Proteus Steinmetz Award from IEEE. He is Fellow of IEEE, EICEJ, and IPSJ, and a member of Television Institute. He has authored six books in the areas of multimedia coding, distribution, and Internet.

Takuyo Kogure holds B.E. degree in electric engineering from Shizuoka University. He spent the first 7 years of his career working on developing electronic components in Matsushita Electric Industrial Co., Ltd. (MEI). Kogure next joined Acoustic Research Lab. at MEI, involved in standard activity of digital audio tape recorder and at the same time, in Magnetic Recording Media Standard in IEC TC 60A. Kogure started to work on MPEG video coding TC area. He next went to Singapore where he started a new MEI laboratory named Audio Video Information Center in Asia Matsushita Electric Co. He was assigned Digital Storage Media (DSM) subgroup Chair of ISO MPEG (SC29/WG11) in 1986, and after that, he kept attending that meeting continuously and now is involved in network security area called IPMP. He returned to MEI Japan, in 1995 and worked on Management committee member of DAVIC, and also works video-on-demand system in MEI. He is one of the founding members of MPEG-4 licensor group formed with MPEG LA. He has also been involved in a variety of national funded projects in Japan, with the aim of contributing MPEG-4 reference bitstreams, implementing MPEG-4 IPMP, and so on.