DOI: 10.3745/KIPSTB.2009.16-B.2.101

선택적 암호화가 가능한 윈도우 미디어 보호 방법

박 지 현[†] · 류 재 철[††]

요 약

IP 셋톱박스에 서비스되는 콘텐츠는 IP 네트워크를 통하여 전송되므로 네트워크 해킹 도구를 이용하면 스트리밍되는 콘텐츠를 저장후 불법이용이 가능하다. 따라서 안전한 스트리밍 서비스를 위해서는 스트리밍 환경에 적합한 콘텐츠 보호 방법이 요구된다. 최근까지 스트리밍 서비스되는 콘텐츠는 MPEG-2 TS를 이용하였지만, 낮은 압축률과 TS 패킷의 높은 오버헤드때문에 현재는 WMV, MPEG-4, H.264로 점차 옮겨가고 있다. 본 논문은 스트리밍되는 윈도우 미디어 콘텐츠를 보호하기 위한 DRM을 제안한다. 제안하는 방법은 기존의 WMV 스트리밍 시스템에 비의존적으로 설계되어 기존의 스트리밍 환경에 적용이 용이하고, 사용자 기기의 성능을 고려하여 DRM 처리 시간을 조절할 수 있도록 선택적 암호화 기법을 제공한다. 본 논문에서는 실험을 통하여 선택적 암호화를 이용하여 DRM 처리시간을 감소시킬 수 있음을 보인다.

키워드 : DRM, 저작권, WMV, 선택적 암호화

Protection of Windows Media Video Providing Selective Encryption

Jihyun Park[†] · Jae-Cheol Ryou[††]

ABSTRACT

As content serviced for IP set-top boxes is streamed over IP network, the existing hacking tools for IP network can be used to capture the streamed content. Until recently, most of the content serviced on IP set-top boxes has been MPEG-2 TS. However, this content will be gradually moved to WMV, MPEG-4 or H.264 because of the relatively low compression efficiency and overhead of the TS packet. In this paper, we propose a DRM scheme other than WMRM for streamed WMV content. Our approach is to design a DRM scheme independent to the existing WMV streaming system. We also design this scheme in order to provide the feature for controlling the DRM processing time considering device performance. We verified it through the experiment.

Keywords : DRM, Copyright, WMV, Selective Encryption

1. Introduction

Streaming refers to the technology used to transmit digital multimedia data over the internet and play them in real time. Whereas content stored in local devices has the problems related to intellectual property rights occurring by illegal copying, streaming content has solved these problems by preventing save functions of client applications. However, several tools have appeared recently that can save the streamed data by intercepting the network packet or by assuming legal players. Thus, streamed media also becomes weak against illegal use.

Several DRM(Digital Rights Management) schemes for downloaded content have been proposed, and some of them support the end-to-end security[1]. However, these schemes cannot support streamed type of content because they change the content file format to what cannot be sent by the streaming server.

Until recently, most of the content serviced on IP set-top boxes has been the streamed MPEG-2 TS[2]. This will be change and be gradually moved to WMV(Windows Media Video), MPEG-4 or H.264 because of the relatively low compression efficiency and overhead of the TS packet

Microsoft's WMRM(Windows Media Rights Manager)[3] is the only DRM for WMV whereas several DRM schemes are applied to other content types like MPEG-2 TS and MP4. It must also be noted that WMRM has difficulties supporting some content providers' own license issuing

[†] 정 회 원 : 한국전자통신연구원 SW 콘텐츠연구부문 선임연구원
[††] 종신회원 : 충남대학교 전기정보통신공학부 교수
논문접수 : 2008년 9월 4일
수 정 일 : 1차 2008년 10월 17일
심사완료 : 2008년 10월 29일

mechanisms because of its strict key scheme. For this reason, a content provider who has already a DRM license server has to construct another WMRM-dedicated license server to service the WMV content. Moreover, it is difficult to design a DRM for WMV on non-Windows systems because MMS(Microsoft Media Server) protocol, the streaming protocol for WMV, as well as WMV are not open technologies.

In this paper, we propose a DRM-based WMV streaming scheme which can not only protect the streamed WMV content but also be easily integrated with the existing WMV streaming system. To integrate our DRM scheme with the WMV streaming system, we encrypt the WMV files without breaking the WMV file format. This enables the encrypted files to be sent by the Windows Media Server as the original WMV files are being sent. We replace the fourCC[4] value in the WMV file with a faked fourCC for our decryption component in order to decrypt the protected content. For a performance issues, our encryption scheme supports selective encryption.

1.1 Related work

WMRM[3] is the DRM technology of Microsoft for protecting Windows media files. WMRM is an end-to-end DRM system that offers content providers and retailers a flexible platform for the secure distribution of digital media files. It provides tools that can be used to package Windows media files and issue licenses for them. WMRM supports the playback on a computer, portable device, or network device. It also supports the persistent content protection among the Windows based devices. It is the only DRM in substance which supports the WMV without changing the file format. A few DRMs support the WMV file, but they changes the file format into their proprietary format. Therefore they only supports the download service but the streaming service. Moreover, they cannot support the existing the media player applications, they can only support the their own player applications.

On WMRM, the content packaging, distribution, and licensing process begins with a piece of encoded content. The content packager packages the content as an encrypted Windows media file and then shares the secrets for decrypting and licensing the file with the license issuer. Consumers are then issued a license to play the file, and use it in accordance with the business rules defined for the content file.

Although this is the easiest way to protect Windows media files, WMRM remains difficult to be customized and to support the interoperability with the other DRM technologies because WMRM is not an open technology. Actual cases in Korea have shown that many DRM providers have been constructing and operating the two types of license server. One is for content protected by its own DRM, and the other is for WMV content.

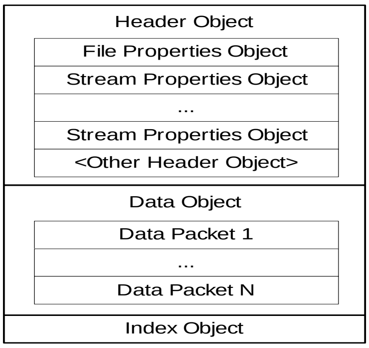
2. Related Technologies

2.1 WMV File Format

WMV file format is based in ASF(Advanced Systems Format)[5] which wraps a video bit stream. ASF is the extensible file storage format developed by Microsoft for authoring, editing, archiving, distributing, streaming, playing, referencing, or otherwise manipulating content. It supports data delivery over a wide variety of networks and is also suitable for local playback

An ASF file is organized into sections called objects. There are three top-level objects, a header object and a data object, and an optional index object. The header object contains all the information that is needed to properly interpret the information within the data object, such as file size, number of streams, error correction methods, and codecs used. Metadata are also stored here. The data object contains all the digital media data for an ASF file. These data are stored in the form of ASF data packets. Each data packet contains data for one or several digital media streams. Data packets are sorted within the data object based on the time when they should be delivered. The index object contains a list of associated timestamp-key frame pairs which enables applications to efficiently seek through a file. (Fig. 1) shows the structure of ASF file.

Most DRM systems use the same encryption method without considering content format, that is the content is protected by encryption of the full data. However, this encryption method cannot be applied to the streaming content. The streaming server can not transmit the stream data correctly since encrypting the full content file changes



(Fig. 1) ASF File Structure

the format of the content. So, the encryption method, which doesn't change the content format, is needed to protect the WMV content and not to modify the streaming server.

2.2 DirectShow

Microsoft DirectShow[6] is architecture for the streaming media on the Microsoft Windows platform. DirectShow provides for high-quality capture and playback of multimedia streams. It supports a wide variety of formats, including WMV, MPEG, AVI, MP3, and WAV sound files. It supports capture using Windows Driver Model(WDM) devices or older Video for Windows devices. DirectShow simplifies media playback, format conversion, and capture tasks. At the same time, it provides access to the underlying stream control architecture for applications that require custom solutions. DirectShow components can be easily extended to support new formats or custom effects.

Microsoft DMOs(DirectX Media Objects)[7] present a new way to write data streaming components. In some respects, DMOs are similar to Microsoft DirectShow filters. Like a DirectShow filter, a DMO takes input data and uses it to produce output data.

Our decryption filter should be implemented in DMO form since Microsoft's windows media player only allows a DMO type filter when it processes the WMV codec related movie files.

3. Proposed WMV Protection scheme

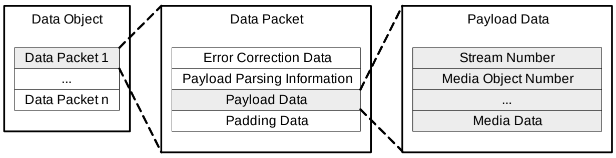
The proposed scheme provides a DRM packaging process and a DRM content consuming process. Any key distribution process and license issuing process can be integrated with this scheme.

3.1 DRM Packaging Scheme

The DRM system for downloaded content encrypts the full file in order to protect it[8]. As mentioned above, the streaming server decides the sending data through information of the movie data as it transmits a movie. If the full file is encrypted for protection, the information needed to read the media data will also be modified. This would make the streaming service impossible.

As not to break the WMV file format, we encrypt only the data packets for video and audio. The other objects included in the WMV file are simply copied to the destination file. The specific process of the proposed scheme is described at the below.

(Fig. 2) shows the detailed structure of the data object in WMV file. The data object consists of many data packets. The data packet is the actual packet which is transmitted to the client device during streaming[5].



(Fig. 2) Structure of Data Object

To preserve the length of each sample, we encrypt the media sample by the size of maximum multiple of 16 bytes which is the same as the 128bit AES[9] block size. The remainder data are not encrypted. The size of these data is less than 16.

When only selected data packets are encrypted, the decrypter component must be able to decide which samples are to be decrypted. We added one byte for indicating the frame encryption - 1 for the encrypted sample, 0 for the not encrypted original sample.

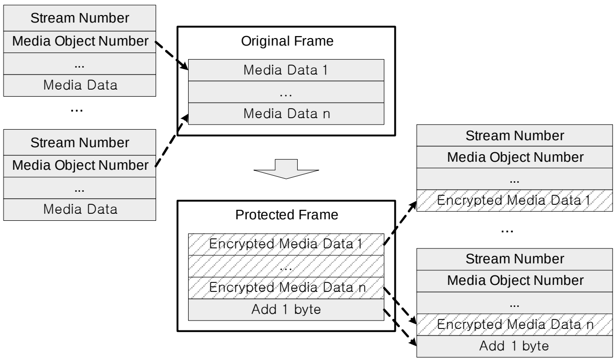
To encrypt the media data in the unit of frame, we aggregate the media data into a frame. It can be achieved by merging the media data of the identical media object number. Then we encrypt the frame data, and attach 1 byte at the tail of the media sample to indicate the encryption. Attaching 1 byte is an optional process only when selective encryption is applied.

With the exception of the last one, the encrypted frame data is divided into the same size of the original media data. The size of the last media data is increased by 1 since we attach 1 byte data to indicate the encryption. (Fig. 3) illustrates this process.

The encrypted media sample C can be represented as the following equation.

$$C = h \mid E_k(M) \mid \mathrm{t} \mid \mathrm{f}$$

h denotes the safe header which is the skipped header



(Fig. 3) Media Data Encryption

of data packet object. Some information in the start of the media sample is used to the decoder. We do not encrypt this part so that the non-decrypted WMV can be played in the screen of the broken pictures. $E_k(M)$ is the encrypted part of the media sample using AES. t denotes the tail that is not encrypted since its length is less than the AES block size. f denotes the optional flag to indicate whether the sample is encrypted or not. This field appears only when the selective encryption is applied. If this field is inserted, some other metadata of WMV should be modified because the file size shall be changed. In the case of encrypting all frames or encrypting all key frames, we can omit this field since we can determine which encryption option is applied by examining the license information. The above encryption scheme is described by the following steps:

Step1. Check media data type

Get a steam number and a stream type from the stream properties object which is located in the header object.

Get a stream number from payload data in a data packet which is located in the data object.

If the two stream numbers are the same, the media type of the data packet is the stream type that was taken from the header.

Step2. Check if a data packet has a key frame

If the MSB(Most Significant Bit) of the stream number is 1, the data packet has a key frame.

Step3. Make one media sample by gathering the packets which have the same media object number.

Step4. Encrypt the media sample by 16 bytes and replace the original data by the encrypted data continuously until the size of the remainder is less than 16 bytes

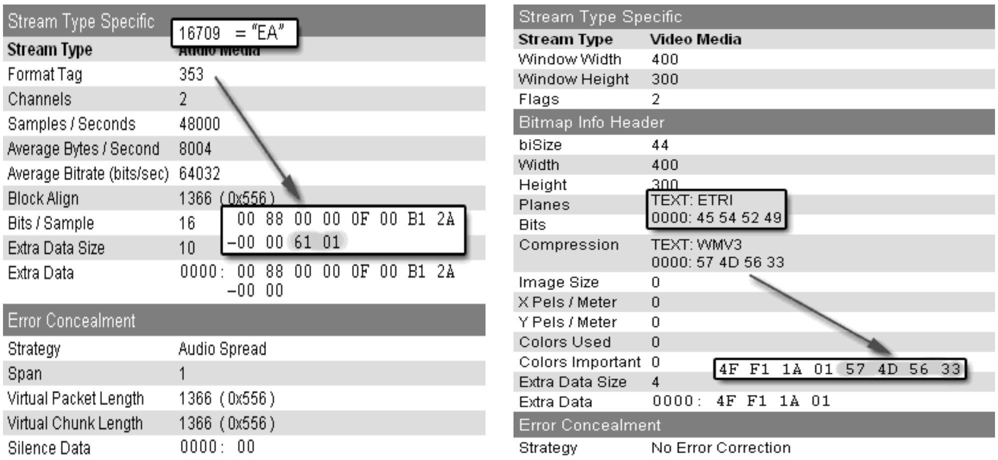
Our encryption scheme provides several encryption options. We distinguish the encrypted frame by 1 byte marking flag. This scheme increases file size, but it makes it possible to encrypt the media data with several options. <Table 1> shows the encryption options of our encryption scheme.

To be loaded properly, our decryption module in the player of a set-top box needs another modification for the WMV file. Automatic decryption by DMO is possible due the fact that the graph rendering system searches the decoder which is capable of decoding media type through the use of specified information in the stream properties object of the WMV file. For usual video files, that leads to proper codec used in the filter graph. But in our case, we should fake the codec ID, which forces a player application to load our custom decrypter instead of the standard codec.

For our purposes, we modify stream properties objects stored in the header object in order to fake the fourCC. Firstly, we replace the codec ID with our decrypter ID, then store the original ID in the tail of extended format block. (Fig. 4) shows the modified stream properties object. In this case, we replace the video fourCC with 'ETRI' which is our custom ID and store the original fourCC in the extra data field of the stream properties object.

<Table 1> Encryption Options

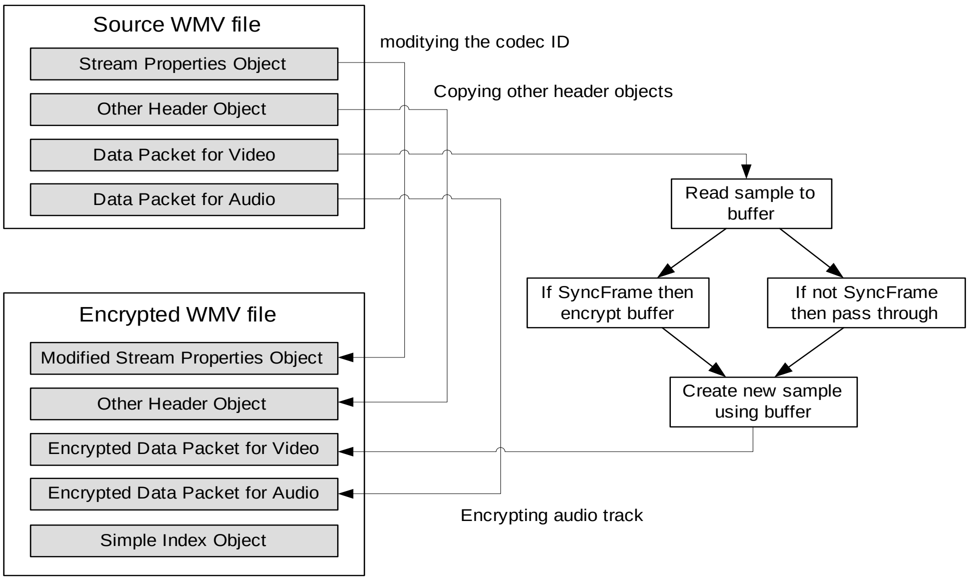
Option Type	Option List
Media Type	- Video Only - Audio Only
For Video Media	- Key Frames Only - Every i-th Frame - All Frames
Range	Time based encryption range



Stream Type Specific	
Stream Type	**Audio Media**
Format Tag	353
Channels	2
Samples / Seconds	48000
Average Bytes / Second	8004
Average Bitrate (bits/sec)	64032
Block Align	1366 (0x556)
Bits / Sample	16
Extra Data Size	10
Extra Data	0000 : 00 88 00 00 0F 00 B1 2A -00 00
Error Concealment	
Strategy	Audio Spread
Span	1
Virtual Packet Length	1366 (0x556)
Virtual Chunk Length	1366 (0x556)
Silence Data	0000 : 00

Stream Type Specific	
Stream Type	**Video Media**
Window Width	400
Window Height	300
Flags	2
Bitmap Info Header	
biSize	44
Width	400
Height	300
Planes	
Bits	
Compression	TEXT: WMV3 0000: 57 4D 56 33
Image Size	0
X Pels / Meter	0
Y Pels / Meter	0
Colors Used	0
Colors Important	0
Extra Data Size	4
Extra Data	0000 : 4F F1 1A 01
Error Concealment	
Strategy	No Error Correction

(Fig. 4) Modified Stream Properties Object



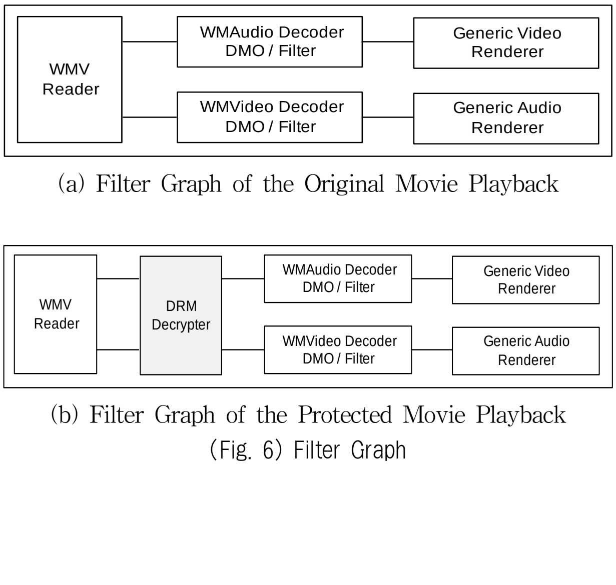
(Fig. 5) Generation of Protected WMV File

(Fig. 5) shows the overall process for generating a protected WMV file briefly.

3.2 DRM Decryption Scheme

As explained in the previous section, we fake the fourCC for Windows graph rendering system to load out decryption module. That alone is insufficient for successful decoding of the video because the output media data processed by our decrypter is still compressed. To overcome this situation, the decrypter must restore the original media types for the outgoing media streams. This allows a filter graph or a client application to put an original decoder just behind of our decrypting filter.

(Fig. 6) shows the difference of two filter graphs - (a) shows the filter graph of the original movie playback, whereas (b) shows the filter graph of the protected movie playback.



(a) Filter Graph of the Original Movie Playback

(b) Filter Graph of the Protected Movie Playback

(Fig. 6) Filter Graph

The decryption process is as follows,

Step1. The decrypter replaces the faked fourCC with the original fourCC of the correct decorder to be connected next to the decrypter

Step2. Get the decryption key from the license information

Step3. Check the last byte of the media sample if the selective encryption is applied

Step4. Decrypt the media sample with the decryption key

Step5. Pass the decrypted media sample to the decoder

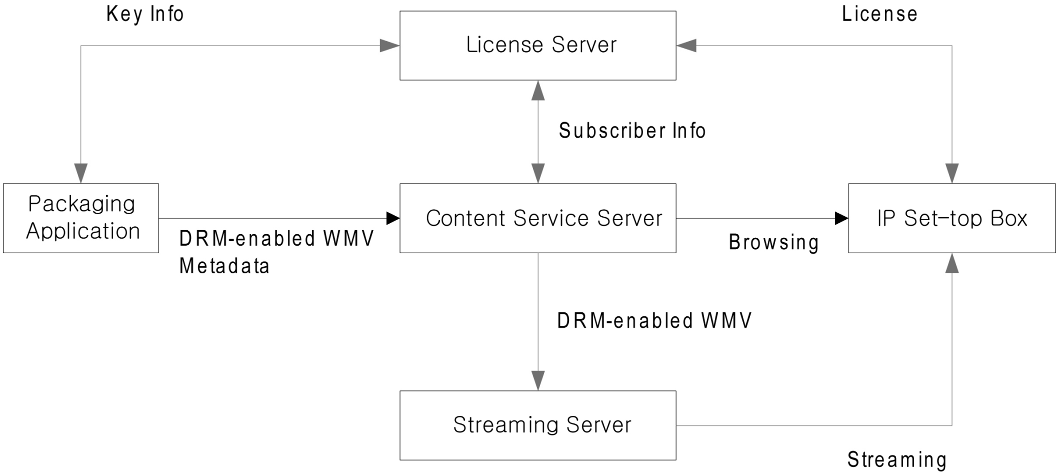
4. Implementation

4.1 System Overview

We implemented a secure WMV service for IP set-top boxes with the proposed scheme. It consists of a packaging application, a key management server, a content service server, and IP set-top boxes as illustrated in (Fig. 7).

The packaging application generates the protected WMV files. It encrypts only the media data part of WMV files in order not to break the WMV file format. We can select the frame type to be encrypted, or the size of the data to be encrypted. The application also sends the information about the encryption to the license server

The license server manages the information needed to decrypt the protected WMV files. It includes key information, the type of the encrypted media data, the size of encrypted data, and so on. The license server sends this information to the legal set-top box.



(Fig. 7) System Components

An IP set-top box includes a media player, a decryption module and a DRM core module. The DRM core manipulates the licenses acquired from the license server. It also securely manages a great deal of authentication information, as well as protects content from being exposed out of our system by authenticating the modules loaded in the client environment.

4.2 Key Distribution

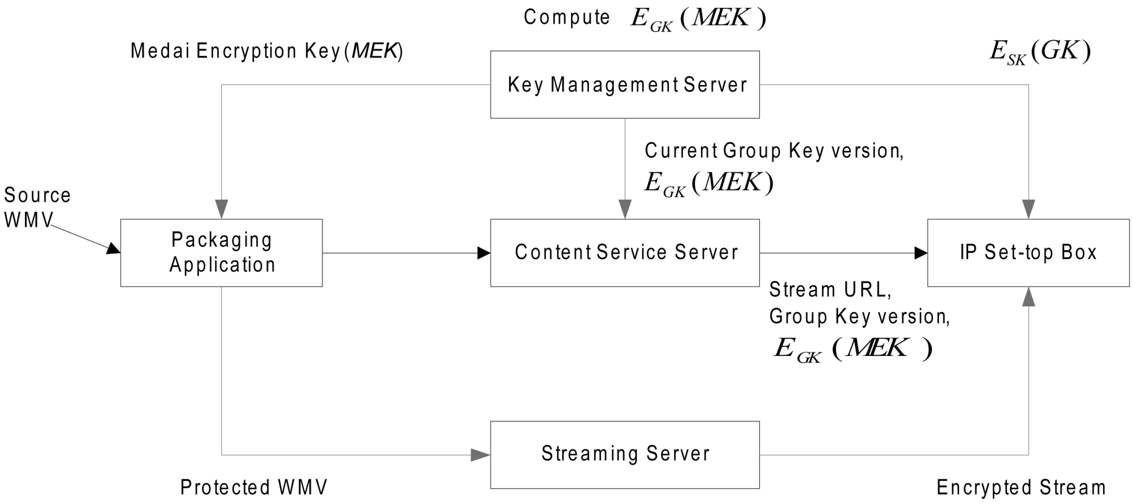
(Fig. 8) shows the key distribution structure of the implemented system. A media encryption key(MEK) is generated by the packaging application and sent to the key management server.

The key management server encrypts MEK with the current group key(GK). If the GK is updated, the key management server encrypts all MEKs with the new GK. On practical service, the GK version can be used for a key updating process. MEK re-encryption is completed before the GK is updated to a new GK. On new GK applied time, it is enough that only this version of new GK is broadcasted. If an IP set-top box receives the GK version and it is different from the current GK version of the device, it gets a new GK from the key management server. The newly issued GK is encrypted with the secret key(SK) of the IP set-top box.

The streaming server sends the requested content to the IP set-top box. The serviced file is in the form of the protected WMV format, which doesn't make any problems because the file corresponds with the WMV file format.

A secret key is issued to an IP set-top box at install time and stored securely. It can be stored into a smart card. The IP set-top box preserves the current GK. The GK is used to decrypt the MEK, and MEK is used to decrypt the streamed data from the streaming server.

<Table 2> describes the kind of keys and their usage.



(Fig. 8) Key Distribution

〈Table 2〉 Key Description

Key	Description
GK	Group key assigned to each user group. This key can be updated periodically. It is used to encrypt MEK.
MEK	Media encryption key to encrypt WMV video and audio data. This key is encrypted with GK and transmitted to set-top boxes from the key management server through the content service server
SK	Secret key of a set-top box. It is issued when a set-top box is installed. GK is transmitted to a set-top box in the form of encrypted valued with SK.

5. Evaluation

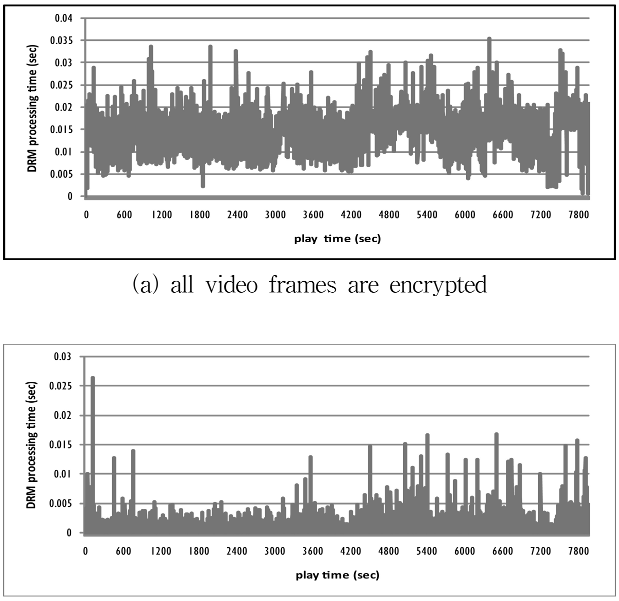
We measured the processing time for DRM on an IP set-top box. Our set-top box has a VIA C3 800Mhz CPU, a 128Mb CF memory, 256Mb RAM and S/W WMV decoder. We used 1Mbps 24fps movie. <Table 3> shows the media data distribution of the test file. We measured the total processing time for DRM every each second.

(Fig. 9) is a chart of DRM performance measurement, (a) is the chart for the file of which all the video frames are encrypted, while (b) is for only key frames encrypted.

For the all frames encrypted file, the average time for DRM is 0.014 sec, the maximum is 0.035 sec, and the minimum is 0.0005 sec. The average decoding time for one frame of 24fps movie is about 41ms. The average DRM processing time for 24 frames data is about 1/3 of the decoding time of one frame. That is, the DRM processing time is the less than 1.5% of the decoding time. Henceforth, we can say that our DRM scheme does not affect content playback time. If selective encryption is applied, the DRM processing time can greatly reduced as shown at the (Fig. 9 (b)) and <Table 4>. The reason is that most of the video samples are not key frames - only about 5% of the video samples are key.

〈Table 3〉 Media Data Distribution of the Sample File

Type	Number	Total Size	Average Size
Video Sample	199,047	819,292,904	4,116
Key Frame	4,601	39,953,716	8,683
Audio Sample	46,689	63,777,174	1,366



(a) all video frames are encrypted

(b) only key frames are encrypted

(Fig. 9) DRM Processing Time Measurement

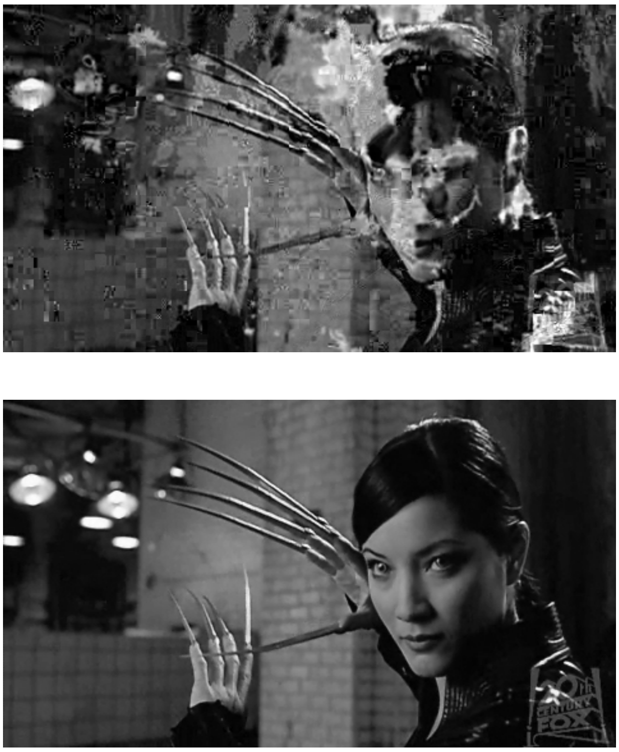
(Fig. 10) shows the difference of two types of playback. The upper image is a screen of the protected WMV file without decryption whereas the lower is with decryption.

6. Conclusion

We designed the DRM functions which are proper to protecting the streamed WMV media on an IP set-top box. With the proposed system, secure WMV streaming

〈Table 4〉 DRM Processing Time per Every Second

Encryption	Max	Min	Average
All Video Samples	0.035s	0.0005s	0.014s
Key Frames only	0.026s	0s	0.0006s



(Fig. 10) Playback Comparison

is possible and can be easily integrated with the current commercial streaming system because the file format is not broken and as before, the existing streaming protocol is used. We replace the original fourCC by our faked fourCC for our decrypter to be loaded automatically by the graph rendering system. To satisfy the performance requirement, our encryption scheme has several encryption options. We can select the amount of encrypted data size considering the streaming performance. This implies that a more complicated encryption algorithm can be applied without degrading the streaming performance. Through experiment, we also showed that the DRM processing time rarely degrades the whole streaming playback.

Acknowledgement

This work was supported by the IT R&D program of MKE/MCST/IITA [2009-S-017-01, Development of user-centric contents protection and distribution technology]. The second author of this research was supported by the Ministry of Knowledge Economy, Korea, under the ITRC support program supervised by the IITA (IITA-2009-(C1090-0902-0016)).

Reference

[1] Y. Jeong, K. Yoon and J. Ryou, “A Trusted Key Management Scheme for Digital Rights Management,” *ETRI Journal*, Vol.27, No.1, pp.114-117, Feb., 2005.

[2] S. Hwang, J. Kim, D. Nam and K. Yoon, “Protection of MPEG-2 Multicast Streaming in IP Set-Top Box Environment,” *ETRI Journal*, Vol.27, No.5, pp.595-607, Oct., 2005.

[3] Windows Media DRM, http://www.microsoft.com/windows/windowsmedia/forpros/drm/default.mspx

[4] FOURCC for Video Compression, http://www.microsoft.com/whdc/archive/fourcc.mspx

[5] ASF Specification, http://www.microsoft.com/windows/windowsmedia/forpros/format/asfspec.aspx

[6] DirectShow, http://msdn2.microsoft.com/en-us/library/ms783323.aspx

[7] DMO, http://msdn2.microsoft.com/en-us/library/ms783356.aspx

[8] Olin Sibert, David Bernstein, and David Van Wie, “DigiBox: A Self-Protecting Container for Information Commerce,” *1st USENIX* Workshop on Electronic Commerce, Jul., 1995.

[9] ADVANCED ENCRYPTION STANDARD (AES), FIPS PUB 197, Nov., 2001.

박 지 현

e-mail : juhyun@etri.re.kr

1997년 서강대학교 전자계산학과(학사)

1999년 서강대학교 컴퓨터공학과(공학석사)

1999년~현 재 한국전자통신연구원 선임 연구원

2005년~현 재 충남대학교 컴퓨터공학과 박사과정

관심분야 : DRM, 정보보호, 멀티미디어, IPTV 등

류 재 철

e-mail : jcryou@home.cnu.ac.kr

1985년 한양대학교 산업공학과(학사)

1988년 Iowa State Univ. 전산학과(석사)

1990년 Northwestern Univ. 전산학과(박사)

1991년~현 재 충남대학교 전기정보통신 공학부 교수

관심분야 : 인터넷 보안, 전자지불시스템