

國立交通大學

電子工程學系 電子研究所碩士班

碩士論文

利用 MPEG-21 REL 實現數位影像廣播系統中
的數位內容保護



**A Digital Content Protection Scheme Using
MPEG-21 REL with Applications to DVB
Systems**

研究生：施瑛姿

指導教授：杭學鳴 博士

中華民國九十四年十一月

利用 MPEG-21 REL 實現數位影像廣播系統中的數位內容保護

A Digital Content Protection Scheme Using MPEG-21 REL

with Applications to DVB systems

研究生：施瑛姿

Student: Ying-Tzu Shih

指導教授：杭學鳴

Advisor: Hsueh-Ming Hang

國立交通大學

電子工程學系 電子研究所碩士班



A Thesis

Submitted to Department of Electronics Engineering & Institute of Electronics

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Electronics Engineering

November 2005

HsinChu, Taiwan, Republic of China

中華民國九十四年十一月

利用 MPEG-21 REL 實現數位影像廣播系統中的數位內容保護

研究生：施瑛姿

指導教授：杭學鳴 博士

國立交通大學
電子工程學系 電子研究所碩士班

摘要

在數位電視中，條件式存取(Conditional Access)提供了設定存取限制的方法。但是一般的條件式存取系只能對數位內容提供存取的限制，並無法更進一步地對接收到的內容之使用權限做保護及控管。本論文的目的 是利用MPEG REL在數位電視系統上實現條件式存取以及權限管理。證明可以 使用MPEG REL作為數位影像廣播系統(DVB)以及數位權限管理系統(DRM) 的橋樑。在本論文中，我們把MPEG REL的License放入至DVB-SI中廣播出去， 使接收端可以根據收到的License來控制使用者的使用權限。於是我們設計 了兩個應用系統來證明透過廣播MPEG REL的License不僅可以做到條件式存 取還可以達到權限管理的功能。另外，在我們的應用中為了達到存取限制， 還設計了一個金鑰管理機制。

A Digital Content Protection Scheme Using MPEG-21 REL with Applications to DVB Systems

Student: Ying-Tzu Shih

Advisor: Dr. Hsueh-Ming Hang

Department of Electronic Engineering & Institute of Electronics
National Chiao Tung University

Abstract

In a digital TV system, the Conditional Access (CA) system is essential to protect digital contents from unauthorized use. However, an ordinary CA system has only the ability to restrict the access to certain material. It cannot provide a further protection and prohibit unauthorized uses after the content has been downloaded and descrambled. The goal of the thesis is to use the MPEG REL to implement both the CA and the rights management systems. We will show how one can use the MPEG REL to bridge the gap between a DVB system and a DRM system. In this thesis, we insert a license, according to MPEG-21 REL expressions, into the Service Information of a DVB bit stream. With this license, the operators can constrain the user's operations on the content through the DRM system and the smart card. To demonstrate the method we propose, we implement two application examples to show that using the REL can achieve both conditional access protection of digital content and manage the utilization of digital assets. In addition to the usage control of the MPEG REL, we also

propose a key management machine in our applications to protect the content from being illegal accessed.



誌謝

很感謝我的指導教授 杭學鳴 老師這兩年來對我的指導，並且提供我機會參與工研院的計畫，讓我學習到許多相關的知識。在生活上，老師也給予我們很多的關懷與鼓勵，支持我度過那些困境，我才能夠持續下去完成我的碩士論文，我真的非常感激。

此外，我也要感謝峰誠學長，他這段日子裡不吝提供經驗，在我論文的準備過程中，多虧有學長的幫助，我才能順利完成。當然，我也非常感謝 Commlab 內一起打拼的夥伴，他們讓我的研究所的生活非常歡樂與溫馨。

最後，我要感謝我的親人以及德原和秀慧。你們的支持與包容，讓我能夠心無旁騖的準備我的論文，在此，謹獻上我最高的謝意。



Contents

| | |
|---|----|
| 摘要 | i |
| Abstract..... | ii |
| Chapter 1 Introduction..... | 1 |
| Chapter 2 Digital Video Broadcasting System | 4 |
| 2.1 Anatomy of an MPEG-2 Stream..... | 4 |
| 2.2 Service Information in DVB systems | 8 |
| 2.2.1 Service Information Description | 9 |
| 2.2.2 Event Information Table | 12 |
| 2.3 Conditional Access Control | 14 |
| Chapter 3 TwinHan DVB-T PCI Card SDK..... | 17 |
| 3.1 Component Object Model | 17 |
| 3.2 DirectShow | 18 |
| 3.3 TwinHan DTV SDK | 19 |
| Chapter 4 MPEG-21 Rights Expression Language specification..... | 24 |
| 4.1 Objectives [7] | 24 |
| 4.2 Data Model | 25 |
| 4.2.1 License..... | 27 |
| 4.2.2 Grant | 27 |
| 4.2.3 Principal..... | 27 |
| 4.2.4 Right | 28 |
| 4.2.5 Resource | 29 |
| 4.2.6 Condition | 30 |
| 4.2.7 Issuer..... | 30 |
| 4.3 Extensibility and Profiling..... | 31 |
| 4.4 Authorization Model..... | 34 |
| 4.5 The Relation with The Other standards in MPEG-21 | 37 |
| Chapter 5 Application Exampmples of the DRM system..... | 38 |
| 5.1 Application One – Digital TV Conditional Access..... | 40 |
| 5.1.1 Cryptography | 41 |
| 5.1.2 Key Management..... | 44 |
| 5.1.3 Architecture Design | 46 |

| | |
|--|----|
| 5.1.4 An Implementation of the Condition Access Scheme | 51 |
| 5.1.5 Conditional Access Demonstration System..... | 55 |
| 5.2 Application 2 –Validity Interval Condition | 59 |
| 5.2.1 MPEG-21 REL Reference Software..... | 60 |
| 5.2.2 Implementation of Validity Interval Condition | 64 |
| 5.2.3 Demo of Validity Interval Condition Scenario | 66 |
| Chapter 6 Conclusion | 72 |
| 6.1 Contributions | 72 |
| 6.2 Future Works | 72 |
| Reference | 74 |



List of Tables

| | |
|--|----|
| Table 2-1 Event Information Section [2]..... | 12 |
| Table 5-1 License Descriptor..... | 38 |
| Table 5-2 Event Information Section [2]..... | 40 |



List of Figures

| | |
|---|----|
| Figure 2-1 Overview of MPEG-2 Transport Stream Multiplexer [22]..... | 5 |
| Figure 2-2 The relationship in the PSI structure..... | 8 |
| Figure 2-3 The structure of a DVB transport stream..... | 9 |
| Figure 2-4 The process of decoding a DVB transport stream [22] | 11 |
| Figure 2-5 The flow of the CA system in a server provider | 14 |
| Figure 2-6 ECM and EMM [3]..... | 15 |
| Figure 2-7 The flow of the CA system in a receiver | 16 |
| Figure 3-1 Filter Graph..... | 18 |
| Figure 3-2 The Whole Frame | 19 |
| Figure 3-3 The Graph Frame | 20 |
| Figure 3-4 The Filter Graph for Transponder Stream..... | 21 |
| Figure 4-1 An example of REL expression | 26 |
| Figure 4-2 Data model of license [6]..... | 26 |
| Figure 4-3 Jane is identified as the holder of a particular key..... | 28 |
| Figure 4-4 The print right | 29 |
| Figure 4-5 An E-book is represented as a digitalResource with a URI..... | 29 |
| Figure 4-6 A constraint condition | 30 |
| Figure 4-7 Issuer..... | 31 |
| Figure 4-8 Extensibility structure..... | 32 |
| Figure 4-9 Authorization model [4]..... | 34 |
| Figure 4-10 The relationships between the entities in the authorization model [11] | 35 |
| Figure 5-1 Cryptography [15] | 42 |
| Figure 5-2 General idea of the conditional access scheme..... | 44 |
| Figure 5-3 Key management procedure | 45 |
| Figure 5-4 Architecture of the conditional access scheme on DVB-T system..... | 47 |
| Figure 5-5 Flow chart of the license server | 50 |
| Figure 5-6 Filter graph of the DVB-T PCI Card in the conditional access scheme . | 52 |
| Figure 5-7 The server provider part and the DVB-T PCI card part of the receiver . | 53 |
| Figure 5-8 Dataflow after retrieving the REL license and the downloaded video file | 54 |
| Figure 5-9 The console window of the first case..... | 58 |

Figure 5-10 Screenshot..... 58

Figure 5-11 The console window of the second case59

Figure 5-12 Dataflow of the reference software [20] 61

Figure 5-13 The control flow of the reference software [21] 64

Figure 5-14 The server provider and the receiver in Application 2..... 65

Figure 5-15 Control flow of rendering and authorization process 66

Figure 5-16 Authorization result dialog 68

Figure 5-17 The screenshot 69

Figure 5-18 Log of authorization 69

Figure 5-19 Authorization result dialog 70

Figure 5-20 Log of authorization 71



Chapter 1

Introduction

Digital Television (DTV), a new type of broadcasting, is quickly catching on around the world. In the DTV system, the TV signals are converted into digital format and transmitted as MPEG-2 transport streams. DTV improves service quality and provides interactivity between consumers and service provider. It also provides better bandwidth utilization than the classical analog color TV broadcasting system. It can also be used to broadcast other types of digital data. Because the digitization of media content can easily create identical copies, how to protect the digital content from unauthorized access and the other illegal uses is an important issue. To solve this problem, the Digital Rights Management (DRM) technology becomes a must.

DRM is a concept for managing and controlling the access and utilization of digital assets. In the DTV system, Conditional Access (or CA) systems have been designed for offering a mechanism to restrict access to certain material (sports or film channels, for example) without permission. The CA systems protect the content intellectual property by encrypting the data so that it can only be accessed by the authorized users. While using CA systems can provide the addressability in a broadcasting network by using the smart card, it can not provide a further protection and manage the authorized uses of the content.

DRM does not mean only access control of digital content but also it includes the right management, which is more flexible in file downloading and content consumption. A DRM system must provide a safe, open, trusty environment for media content delivery and must provide interoperable service. A key issue when trying to

implement interoperable applications for managing and distributing multimedia content is the consideration of their digital rights. A recently approved ISO standard, MPEG-21 Part 5 - Rights Expression Language (REL) [4], is an important technology for the development of interoperability across various DRM systems. The MPEG REL is an XML-based language that can declare rights and conditions for the authorized distribution and use of any content, resources, or services. It is intended to facilitate the expression and representation of rights regarding the digital resources and support guaranteed end-to-end interoperability, consistency and reliability between different systems and services.

The goal of the thesis is to study the MPEG-21 REL standard, use it to implement both CA and rights right management systems. We will show how one can use the REL as the bridging between a Digital Video Broadcast (DVB) system and Digital Rights Management (DRM) system. The DRM system protects digital content by associate it with usage rules. In this thesis, we insert a license, according to MPEG-21 REL expressions, into the Service Information in the DVB system. With this license, the operators can constrain the users' usage on the content through the DRM player and the smart card. To demonstrate the method we propose in practice, we implement two application examples to show that using the REL can achieve both the conditional access protection of digital content and the utilization management of digital assets. Besides the use of the MPEG REL, we also propose a (encryption) key management machine in one of our applications to protect the content from illegal access.

This thesis is organized as follows. In chapter 2, we introduce the DVB digital TV system including of the specification for Service Information in DVB systems and the Conditional Access system of DTV. Chapter 3 briefly describes the SDK of the

Twinhan DVB-T PCI Card, which is used in our experiments to make the demonstration closer to practical system. Next, we will give an overview of the MPEG REL in chapter 4. In chapter 5, we describe the details of our design and implementation of the content protection scheme for the Terrestrial DVB system by using MPEG-21 REL concept an format. A briefly conclusion is given in chapter 6.



Chapter 2

Digital Video Broadcasting System

In 1995, the Digital Video Broadcasting (DVB) project organized by the European Broadcasting Union (EBU) has published a set of formal standards which define a new Digital Video Broadcast system. These standards are widely used in Europe, Asia, Australia, and many other regions of the world starting in 1996. Parts of DVB standards describe digital TV transmission over satellite (DVB-S; EN 300 421), cable (DVB-C; EN 300 429), and terrestrial systems (DVB-T; EN 300 744).

A digital TV program is transmitted as a bit stream of MPEG-2 data known as a transport stream. Although the DVB-T system is used in the implementation examples of the thesis, the source coding and multiplexing approach can also be used in the satellite and cable systems. Here we focus on the structure of a DVB transport stream and the service information in the DVB system. Additionally, we will also describe the general model of Digital TV Condition Access mechanism.

2.1 Anatomy of an MPEG-2 Stream

In any digital broadcasting system, the compression technique used is the MPEG-2 video/audio, and the signal format is the MPEG-2 transport stream (TS). The compressed video and audio contents are called elementary streams (Video ES and Audio ES; ES), each containing one video or one audio track. To make the multiplexing process easier, these elementary streams are split into packets, called packetized elementary stream or PES. The MPEG-2 standard, ISO/IEC 13818-1[1], defines a multiplexing system which combines Video/Audio or data packetized

elementary streams to form a transport stream. The multiplexing system also allows one or more transport streams to be combined into a transport stream.

To create a transport stream, the PES streams are packaged into transport stream packets, which have a fixed size of 188 bytes and consist of 4 bytes header and 184 bytes data payload. This extra level of packetization allows this format to support much more powerful error collecting techniques.

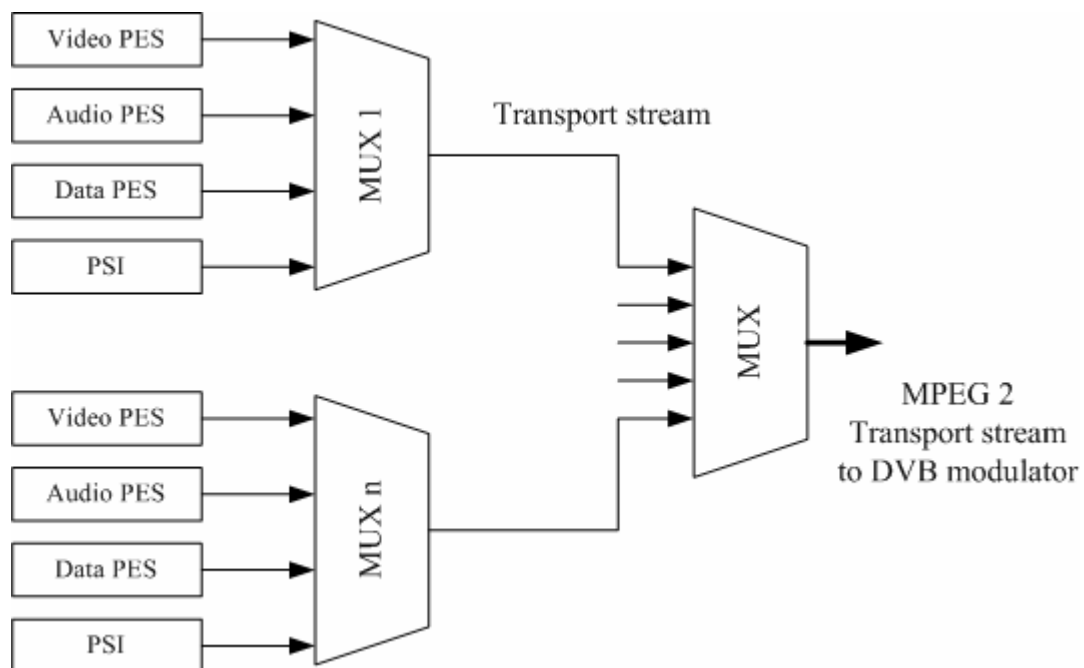


Figure 2-1 Overview of MPEG-2 Transport Stream Multiplexer [22]

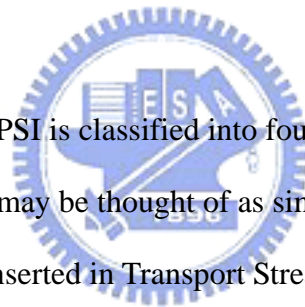
As shown in Figure 2-1, in a DVB system, the multiplexing technique is used to put more than one programs into one single transport stream, which has a data rate of around 25 megabits/second for terrestrial networks. Finally, the transport stream is the output sent to a DVB modulator. A modulator will convert it to an analog signal suitable for broadcasting.

Because a transport stream can carry multiple elementary streams, there must be a way to reconstruct these streams into something that a receiver can present to users.

To solve this problem, MPEG and DVB both specify that other information should be added into the transport stream and this data are coded in a number of elementary streams which are added into the transport stream during the multiplexing process. The data defined by MPEG is known as Program Specific Information (PSI). This PSI can be thought of a number of tables in which each describes the structure of the transport stream and the type of data contained in the stream.

In the multiplexing system, each elementary stream has a unique Packet ID (PID) which is contained in the header of transport stream packets. The PID identifies, via the PSI tables, the contents of the data contained in the Transport Stream packet. Transport Stream packets of the same PID value carry data of one and only one elementary stream.

In the Transport Stream, PSI is classified into four table structures as shown below. While these structures may be thought of as simple tables, they shall be segmented into sections and inserted in Transport Stream packets.



1) Program Association Table (PAT)

Within a transport stream, each group of elementary streams that makes up a single TV channel is a program, or called a service in DVB system. The PAT indicates the location (the PID value of the transport stream packets) of the corresponding Program Map Table of each service. The PAT also gives the location of the Network Information Table. The PAT is always assigned to PID 0x000. Each Transport Stream shall contain one PAT.

2) Program Map Table (PMT)

The PMT describes the locations and type of the streams used by a service, and it tells the receiver the location of the Program Clock Reference fields for a service. The PMT is not broadcast on a fixed PID. For each service in a transport stream, there must be a corresponding PMT within the transport stream.

3) Conditional Access Table (CAT)

The CAT provides the information about the CA systems that are in use in the transport stream. The information tells the receiver how to decode them; it is private and depend on the CA system. The CAT is sent with the well-known PID value of 0x001.

4) Network Information Tables (NIT)

The NIT provides information about the physical network and also describes how transport streams are organized on the current network. The data format is outside the scope of ISO/IEC 13818-1 [1]. The syntax and semantics of the NIT are defined in a specification of DVB [2].

Figure 2-2 is a simple example to present the relationship between the four PSI tables and the Transport Stream.

In DVB system, the PSI is referred to as Service Information (SI). Besides these four tables listed above, DVB specifies other nine tables in the Service Information standard [2].

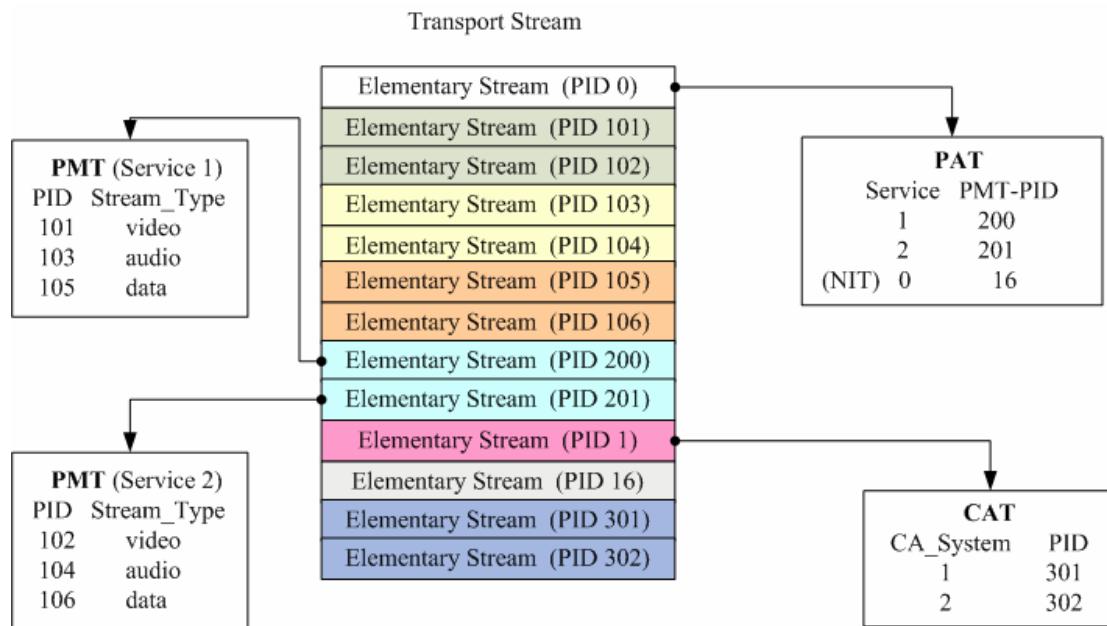


Figure 2-2 The relationship in the PSI structure

2.2 Service Information in DVB systems

The structure of a DVB transport stream is shown in Figure 2-3. In the DTV-terms, a transport stream is also known as a multiplex, because it contains a number of services multiplexed together. Each service consists of a group of elementary streams that makes up a single TV channel. Each TV show is known as an Event, which contains at least one audio stream, at least one video stream and usually several data streams. The service (TV channel) consists of a series of individual events broadcast one after another.

In addition to the PSI defined by MPEG, DVB specifies other information to provide identification of services and available events for the user. The coding rule of this data is defined in ETSI EN 300 468.

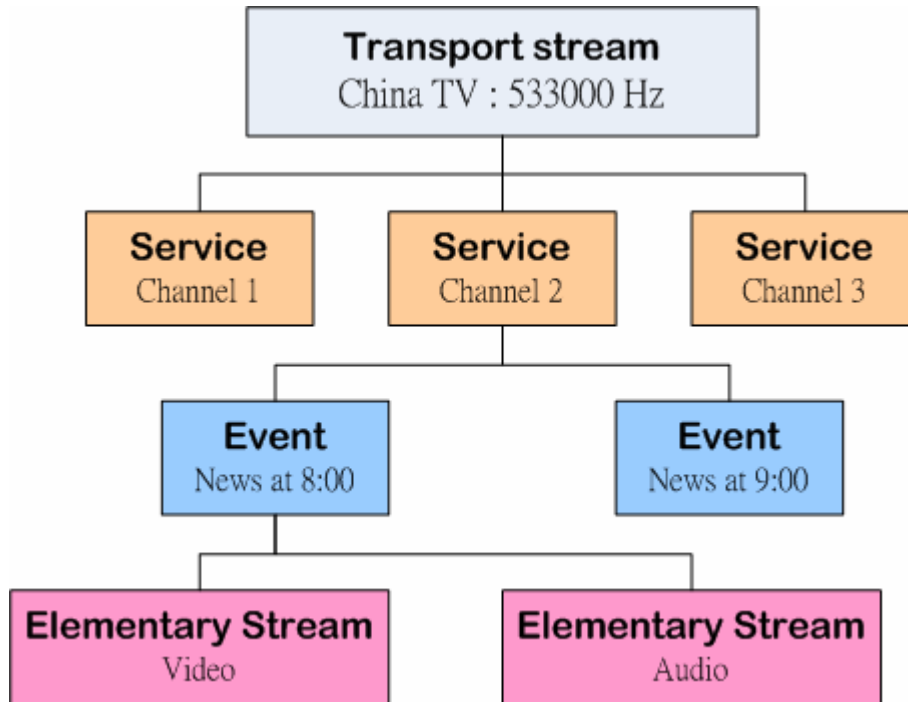


Figure 2-3 The structure of a DVB transport stream

2.2.1 Service Information Description

In contrast to the PAT, CAT, and PMT of the PSI, which give information only about the architecture of the multiplex for helping the receiver to reconstruct these streams, the additional information defined in EN 300 468 can also provide information on services and events carried by different multiplexes.

This data is known as Service Information (SI). The SI data is structured as nine types of tables [2]:

1) Bouquet Association Table (BAT) provides information regarding bouquets. A bouquet is several services that are grouped together logically, for instance, sports package (which contains 8 sports channels) or movie package (which contains 5 movie channels). As well as giving the name of the bouquet, it provides a list of services for each bouquet.

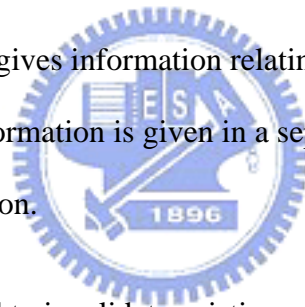
2) Service Description Table (SDT) contains data describing the services in the system e.g. names of services, the service provider, etc.

3) Event Information Table (EIT) contains data concerning events or programs such as event name, start time, duration, etc.

4) Running Status Table (RST) gives the status of an event (running/not running). The RST updates this information and allows timely automatic switching to events.

5) Time and Date Table (TDT) gives information relating to the present time and date. This information is given in a separate table due to the frequent updating of this information.

6) Time Offset Table (TOT) gives information relating to the present time and date and local time offset. This information is given in a separate table due to the frequent updating of the time information.



7) Stuffing Table (ST) is used to invalidate existing sections, for example at delivery system boundaries.

8) Selection Information Table (SIT) is used only in "partial" (i.e. recorded) bitstreams. It carries a summary of the SI information required to describe the streams in the partial bitstream.

9) Discontinuity Information Table (DIT) is used only in "partial" (i.e. recorded) bitstreams. It is inserted where the SI information in the partial bitstream may be discontinuous.

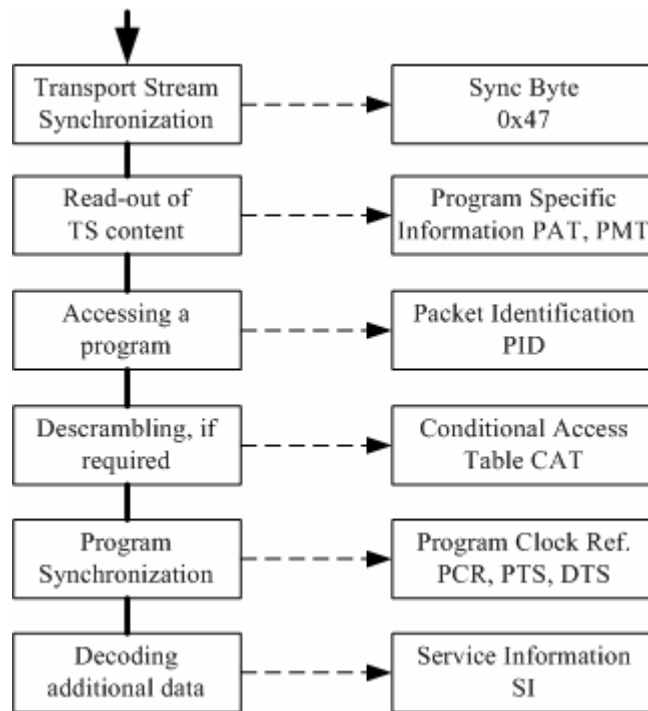


Figure 2-4 The process of decoding a DVB transport stream [22]

A prototypical decoder for a DVB transport stream in the system layer is depicted in Figure 2-4 to illustrate how to construct these streams. In the process of decoding, synchronization is the first step. The sync byte is 0x47. After that, decoders extract video, audio and data streams according to PAT and PMT. If the transport stream is scrambled, the CAT is used to descramble it. Then these elementary streams are synchronized in according with some parameters which are added in the process of MPEG-2 encoding. These parameters include PCR (Program Clock Reference), PTS (Program Time Stamp) and DTS (Decoding Time Stamp). If there are SI tables in the transport stream, decoders must decode the content in these tables.

Using the SI tables, more applications could be developed in DVB system. In this thesis, we just use the Event Information Table to implement a DRM application. Next, we will describe the definition and syntax of EIT.

2.2.2 Event Information Table

The EIT provides information in chronological order regarding the events contained within each service. For mapping all MPEG-2 tables and SI tables into transport stream packets, these tables shall be segmented into one or more sections. So that, the EIT must be segmented into one or more Event Information Sections, before insertion into Transport Stream packets, with the following syntax.

| Syntax | No. of Bits | Identifier |
|------------------------------|-------------|------------|
| event_information_section(){ | | |
| table_id | 8 | uimsbf |
| section_syntax_indicator | 1 | bslbf |
| reserved_future_use | 1 | bslbf |
| reserved | 2 | bslbf |
| section_length | 12 | uimsbf |
| service_id | 16 | uimsbf |
| reserved | 2 | bslbf |
| version_number | 5 | uimsbf |
| current_next_indicator | 1 | bslbf |
| section_number | 8 | uimsbf |
| last_section_number | 8 | uimsbf |
| transport_stream_id | 16 | uimsbf |
| original_network_id | 16 | uimsbf |
| segment_last_section_number | 8 | uimsbf |
| last_table_id | 8 | uimsbf |
| for(i=0;i<N;i++){ | | |
| event_id | 16 | uimsbf |
| start_time | 40 | bslbf |
| duration | 24 | uimsbf |
| running_status | 3 | uimsbf |
| free_CA_mode | 1 | bslbf |
| descriptors_loop_length | 12 | uimsbf |
| for(i=0;i<N;i++){ | | |
| descriptor() | | |
| } | | |
| } | | |
| CRC_32 | 32 | rpchof |
| } | | |

Table 2-1 Event Information Section [2]

This EIT sections shall be transmitted in TS packets with a PID value of 0x0012. The EIT section is identified by the use of **table_id** which is an 8-bit field and always assigned 0x4E. The bits of the next three fields (**section_syntax_indicator**, **reserved_future_use** and **reversed**) shall be set to 1. The **section_length** specifies

the number of bytes of the section, starting immediately following the `section_length` field and including the CRC. The `section_length` shall not exceed 4093 so that the entire section has a maximum length of 4096 bytes. The **service_id** is a label used to identify this service from other service within a transport stream. The **version_number** and **current_next_indicator** indicate that which version of EIT section is valid. The **section_number**, **next_section_number**, **segment_last_section_number** and **last_table_id** allows receiver reassembling these sections. **transport_stream_id** serves as a label for identification of this transport stream from any other multiplex within the delivery system. **original_network_id** gives the label identifying the `network_id` of the originating delivery system.

The inside of the for-loop contains the essential information about the TV shows (events) of a service. The **event_id** is the unique identification number of the described event. The **start_time** and **duration** just are the start time and duration of the event respectively. The **running_status** indicates the status of the event: running or not, pausing and stats in a few seconds. All the component streams of the event are scrambled or not are indicated in the **free_CA_mode** field. The **descriptors_loop_length** gives the total length in bytes of the following descriptors. The **CRC_32** field contains the Cyclic Redundancy Check value.

Each descriptor loop contains one or more descriptors contain some information for the decrypted event. ETSI defined many descriptors that can be re-used in the SI tables. This allows a flexible approach to the organization of the tables and allows for future compatible extensions. For all descriptor, the **descriptor_tag** and **descriptor_length** are applied to it for identifying each descriptor and specifying the length respectively. ETSI also defined some `descriptor_tag` values to allow user defining

these value for their requirement. By using different descriptors, the EIT can transmit different kinds of event information.

2.3 Conditional Access Control

Conditional access (or CA) system is designed for offering the ability to restrict access to certain material (sports or film channels, for example). To each CA vendor, the techniques and implementations used for this is proprietary. In general, the CA model used in the DVB system includes three main functions: scrambling / descrambling, entitlement checking and entitlement management. Figure 2-5 is the general model of digital TV Conditional Access in a service provider.

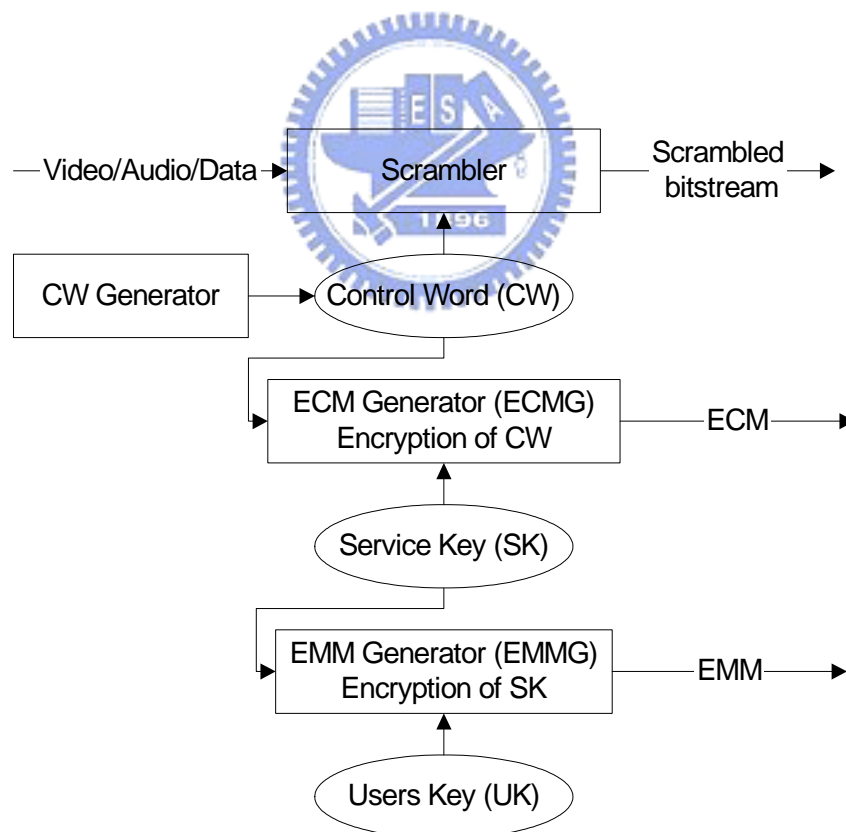


Figure 2-5 The flow of the CA system in a server provider

The control word (CW), which provides the first level of scrambling the TV content, is encrypted using the service key (SK). This service key may be common to a group of users. The encrypted control word is broadcast in an Entitlement Control Messages (ECM) which is shown in Figure 2-6. Next, for making sure that only authorized users can decrypt the control word, the service key is encrypted using the user key (UK) which is unique to each single user. For each authorized user, the service key encrypted using their user key is broadcast as part of an EMM Entitlement Management Messages (EMM) which is shown in Figure 2-6.

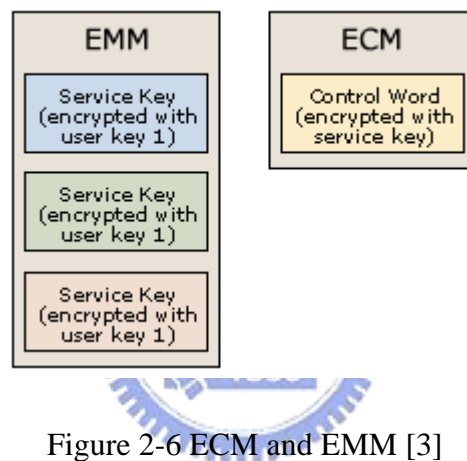


Figure 2-6 ECM and EMM [3]

These two private CA messages, ECM and EMM, must also be sent in transport stream packets to allow the receivers descrambling the TV content before decoding. Entitlement Control Messages specify control words and possibly other, typically stream-specific, scrambling and/or control parameters. Entitlement Management Messages specify the authorization levels or the services of specific decoders. They may be addressed to single decoders or groups of decoders [1]. The conditional access descriptor listed in the Conditional Access Table (CAT) is used to specify both EMMs and ECMs.

Figure 2-7 presents the general CA model for descrambling the TV content. When the receiver gets a CA message, it is passed to the CA system. For authorized

users, the receivers use their user key to decrypt the service key in EMM. Then, the receivers initialize the descrambling hardware and actually descramble the content after decryption of control words in ECM.

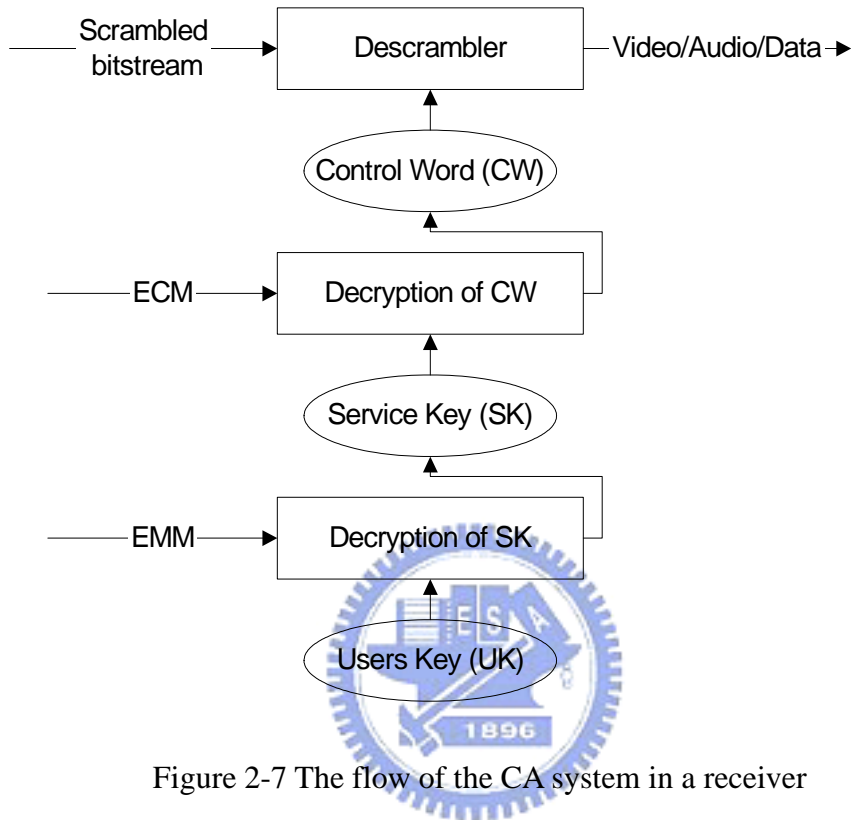


Figure 2-7 The flow of the CA system in a receiver

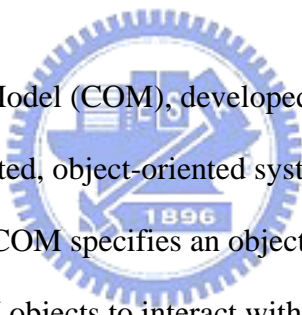
Not all CA systems use the same algorithms. There are some variances between them. And the EMMs may be used for other CA-related tasks besides decrypting service keys. But, basically, they work in the same way.

Chapter 3

TwinHan DVB-T PCI Card SDK

In this thesis, we use the DVB-T PCI Card made by the TwinHan company to receive and de-multiplex the DVB- Terrestrial signal. The Software Development Kit (SDK) of the Twinhan DVB-T PCI Card (called Twinhan DTV SDK for short) will be introduced in this chapter. The SDK is written in C++ and based on the Component Object Model (COM). Microsoft DirectX 8.1 or above is requirement for using the card to watch digital TV.

3.1 Component Object Model



The Component Object Model (COM), developed by Microsoft, is a platform-independent, distributed, object-oriented system for creating binary software components that can interact. COM specifies an object model and programming requirements that enable COM objects to interact with other objects. COM objects are discrete components, each with a unique class identity (called CLSID). They are completely language-independent and have built-in inter-process communications capability and easily fit into an object-oriented program design. Conceptually, a COM objects can be considered as a black box that can be used by applications to perform one or more tasks without getting any idea of how it works. They are most commonly implemented as a dynamic-link library (DLL).

Accessing to an object's data is achieved exclusively through one or more sets of related functions. These function sets are called interfaces, and the functions of an interface are called methods. Each interface is identified by a unique identity.

Communication between two objects in a system occurs by calling the functions in an interface through a pointer to that interface.

3.2 DirectShow

The TwinHan DTV SDK uses some components provided by DirectShow. DirectShow is one of components in Microsoft DirectX. It is designed to simplify the task of creating digital media applications on the Windows platform by isolating applications from the complexities of data transports, hardware differences, and synchronization.

DirectShow uses a modular architecture, where each stage of processing is done by a COM object called a filter. As Figure 3-1 shows, each filter is connected to one or more other filters. The filter chain is called a filter graph. The connection points are also COM objects, called pins. The pins are divided into two classes, output and input pin. Filters use pins to move data from one filter to the next. The arrows in Figure 3-1 show the direction in which the data travels. DirectShow also provides a COM object, Filter Graph Manager, which could control the filters in a filter graph.

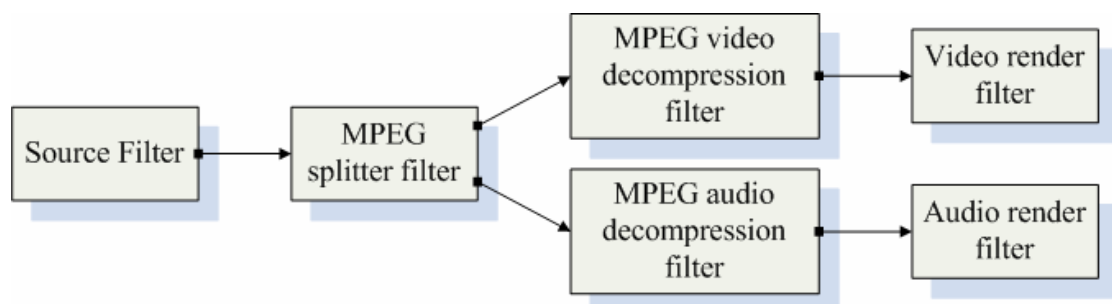


Figure 3-1 Filter Graph

3.3 TwinHan DTV SDK

This section explains the concept and architecture of Twinhan DTV SDK for windows. It describes how to use the SDK to receive, process, and render the DVB signals. Figure 3-2 is the whole frame of the Twinhan DVB-T PCI Card.

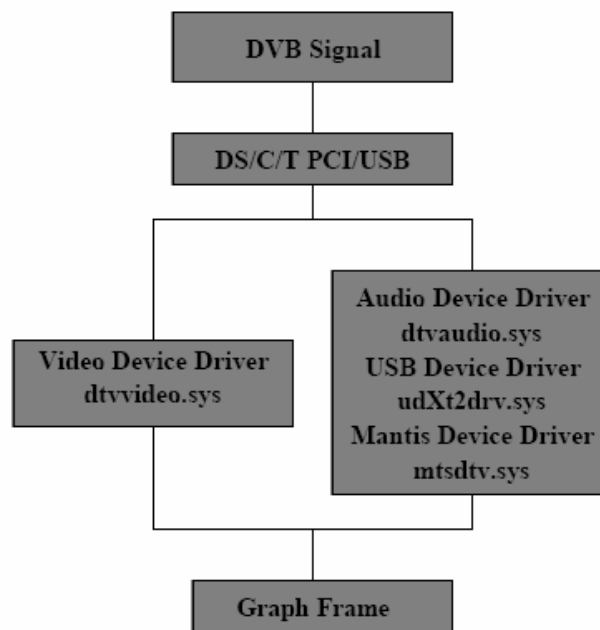


Figure 3-2 The Whole Frame

Twinhan DTV SDK provides COM objects and the interfaces that assist in demultiplexing, decoding, and other signal processing. Using Twinhan DTV SDK simplifies the signal processing procedure. Figure 3-3 illustrates the high-level architecture of the SDK. It is the procedure of rendering the source signals.

The SDK offers a DVB Source filter which should only be loaded through a graph manager which will handle all the pin connections and provide access to the different decoder interfaces. It has four output pins for “Stream 1”, “Transponder Stream”, “Analog Video”, and “VBI”. The connection format of “Stream 1” output

pin is MEDIATYPE_Stream/MEDIASUBTYPE_NULL. It works in the pull mode. When it is working, the PES stream which only includes video and audio will flow out. In order to work properly, before connecting the output pin to the next filter, we should initialize the filter with the open interface. The connection format of another pin “Transpond Stream” is MEDIATYPE_Stream/MEDIASUBTYPE_MPEG2_TRANSPORT, and it is working in the push mode. When it is active, the whole transport stream will flow out from it. The output pin “Analog Video” is designed for Analog Video. Its connection format is MEDIATYPE_Video/MEDIASUBTYPE_RGBx and it use Source Stream as the output pin type. For the output pin “VBI”, the connection type is MEDIATYPE_VBI/MEDIASUBTYPE_VPVBI. It delivers a stream of VBI waveform samples.

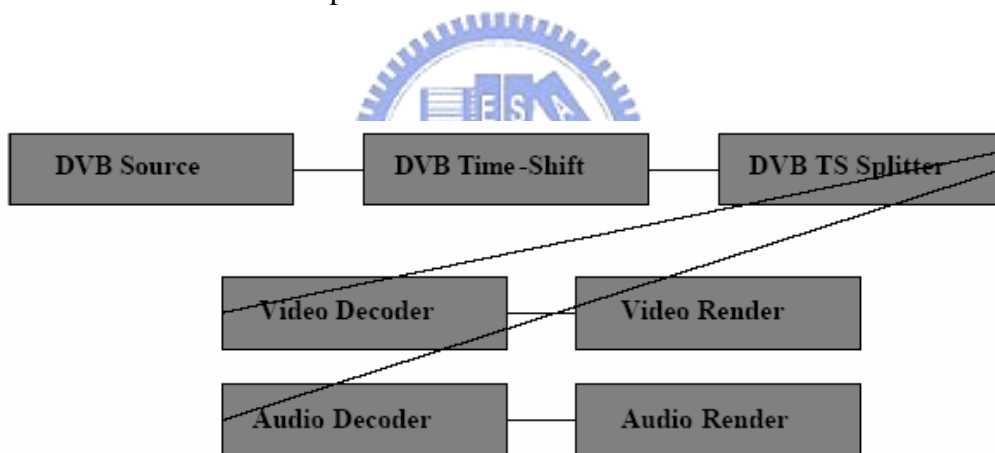


Figure 3-3 The Graph Frame

The TS Splitter filter takes in the interleaved stream and outputs separate audio and video streams. It has 1 input pin and 2 output pins, one is for video, which supports MPEG2_Video; another is for audio, which supports MPEG1AudioPayload and AC3. If the stream includes AC3 audio, the connection format is DOLBY_AC3.

The DVB Time-Shift filter allows seeking and positioning in a previously encoded MPEG2 stream. The special functionality of the DVB Time-Shift filter

allows data seeking with real-time multiplexed streams. DVB Time-Shift filter has 1 input pin and 1 output pin, it needs to be linked between the source filter and the splitter. User only controls the graph state for the time-shifting function, such as pause for enter time-shifting mode, play for playback in time-shifting mode and stop for exit.

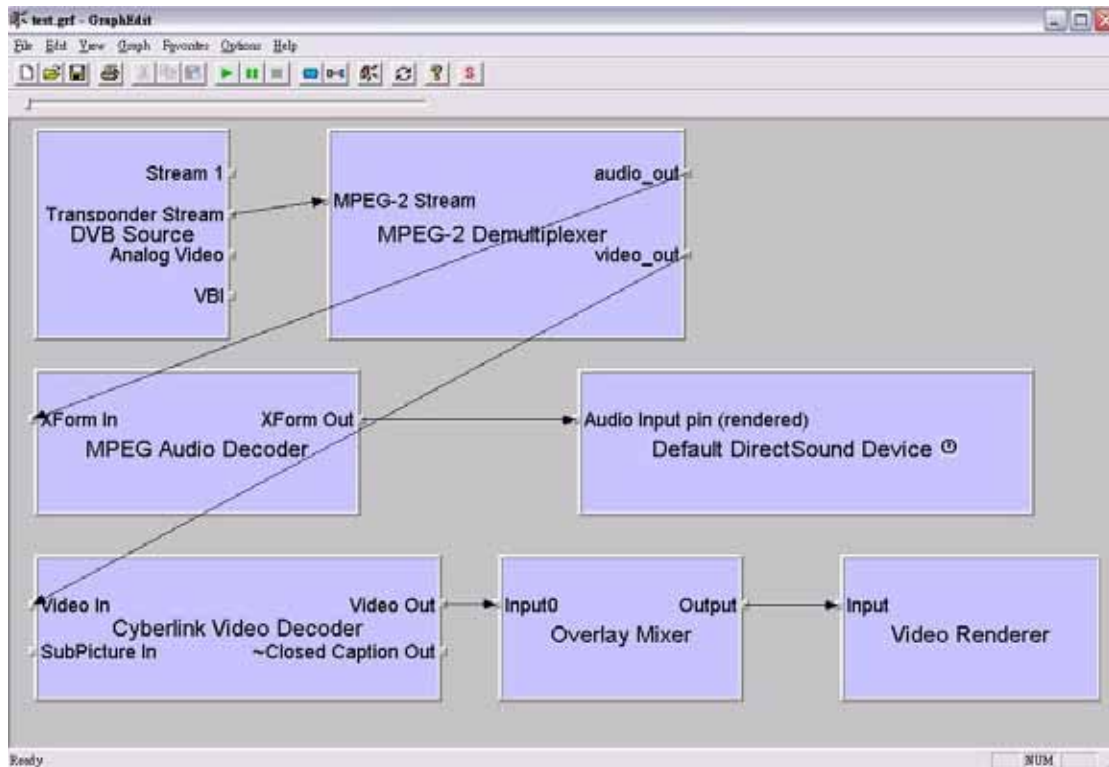


Figure 3-4 The Filter Graph for Transponder Stream

Figure 3-4 is the Filter Graph for receiving a real-time DVB-T signal. Here we focus the DVB Source filter because all the filters in Figure 3-4 are provided by DirectShow except the Source Filter. For more information on these filters refer to DirectShow. The Source Filter will detect the card automatically and work well on this card. It supports more than one interface to allow user to:

- Change the transponder and other options, such as LNB.
- Lock the channel.

- Access to multi-card and support the media files (MPEG2 format) playback function.
- Read remote control code and turn on/off the tuner power.

Using the interface, the user can lock a channel, capture PES stream which includes video and audio streams, scan programs etc. This is achieved through some supported APIs as follows.

STDMETHOD(LockChannel)

```
( THIS_
    long Frequency,          //[in] frequency, KHz
    long SymbolRate,        //[in] bandwidth
    int HV,                  //[in] 1 for Digital, 0 for Analog
) PURE;
```

STDMETHOD(SetPid)

```
( THIS_
    int SetType,             //[in] type for control stream
    long VideoPid1,          //[in] video stream pid
    long AudioPid1,         //[in] audio stream pid
    bool Scrambled,         //[in] the program is scrambled or not
    long PMTPid,             //[in] if Scrambled is valid, it must be filled
    long ProgramNumber,     //[in] if Scrambled is valid, it must be filled
    long VideoPid2,
    long AudioPid2
) PURE;
```



STDMETHOD(ScanChannel) () PURE;

In order to play back programs, user must call two APIs, “LockChannel” and “ScanCannel”, to lock and scan the channel that the user wants to play firstly, and then use the API “SetPid” to set the PID of video and audio for one program. When all are done, the stream will be ready and user can render the output pin to build the

whole graph. Besides these three APIs, there are also plenty of APIs supported to let the Source Filter more powerful.

Note that all the filters, such as demultiplexer, video and audio decoder need to be loaded in the graph prior to rendering graph. Because the formats of the video and audio may be distinct between different programs, it needs re-render the whole graph when changing channel in this frame now.

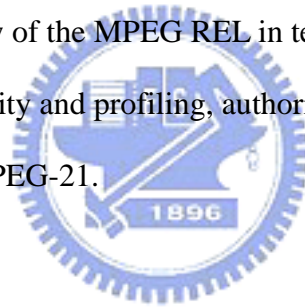


Chapter 4

MPEG-21 Rights Expression Language specification

In this chapter, we will briefly review the International Standard ISO/IEC 21000-5:2004, MPEG-21 Rights Expression Language [4] (called the MPEG REL for short). The ISO/IEC MPEG, Moving Picture Experts Group, committee started the development process for the MPEG REL in 2001 with a Call for Proposals and an evaluation process. After a few years of efforts, the Extensible Rights Markup Language (XrML) proposed by ContentGuard [5], was selected as the core architecture and base technology [6].

We will give an overview of the MPEG REL in terms of its objectives, data model, structure for extensibility and profiling, authorization model, and the relation with the other standards in MPEG-21.



4.1 Objectives [7]

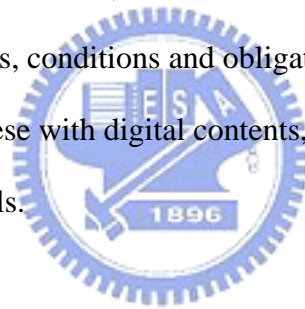
The MPEG REL is an XML-based declarative language that can declare rights and conditions for authorized distribution and use of any content, resources, or services. It defines the syntax and semantics of a machine interpretable language that can be used to specify rights unambiguously.

The MPEG REL is intended to provide a flexible, interoperable mechanism to support transparent and augmented use of digital resources in publishing, distributing, and consuming digital content (including digital movies, digital music, electronic books, broadcasting, interactive games, computer software and other creations in digital form) in a way that protects digital content and honors the rights, conditions,

and fees specified for digital contents. It is also intended to support specifications of access and to control digital content usage in the cases where financial exchange is not part of the terms of use, and to support exchange of sensitive or private digital contents.

For protecting consumer privacy, the MPEG REL provides a flexible interoperable mechanism to ensure personal data is processed in accordance with the individual rights and to meet the requirements for users to express their rights and interests in a way that addresses issues of privacy and use of personal data.

To support guaranteed end-to-end interoperability, consistency and reliability between different systems and services, the MPEG REL must offer richness and extensibility in declaring rights, conditions and obligations, ease and persistence in identifying and associating these with digital contents, and flexibility in supporting multiple usage/business models.



4.2 Data Model

The main function of the MPEG REL is to specify rights related to digital resources (such as content, services, or software applications). Using this language, anyone owning or distributing digital resources can identify

- The principals (such as users, groups, devices, and systems) who are authorized to use the resources
- The rights accorded to those principals
- The condition that must be met before the right can be exercised

```

license
  grant
    Jane
    Print
    E-book "Why Cats Sleep and We don't"
    3 times
  issuer
    T Publisher

```

Figure 4-1 An example of REL expression

For example, an E-book named “Why Cats Sleep and We don’t”, distributed by “T Publisher”, to a consumer (Jane) that she is allowed to print 3 times. Figure 4-1 shows this simple REL example, Jane is granted the right to print the E-book “Why Cats Sleep and We don’t” 3 times under the permission of T Publisher, in its structural form.

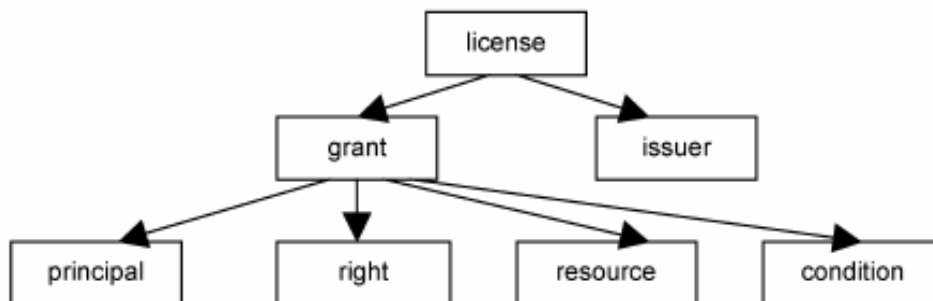


Figure 4-2 Data model of license [6]

In the MPEG REL terminology, Jane is considered as a *principal*; print, a *right*; E-book “Why Cats Sleep and We don’t”, a *resource*; 3 times, a *condition*; and T Publisher, an *issuer* of the right. This simple example above is consists of the essential elements and the relationship among those elements of the MPEG REL expression. The expression conveys a statement of the following form: An issuer states that a principal has some right to a resource under some condition. The right-granting

portion of this statement is called a *grant* and the entire statement is called a *license* which contains both grants and issuers. In this example, the statement, Jane can print the E-book “Why Cats Sleep and We don’t” 3 times, is a grant; the statement, T publisher issues a grant that Jane can print the E-book “Why Cats Sleep and We don’t” 3 times, is a license.

The essential elements and the relationship among those elements of the MPEG REL are shown in Figure 4-2, the MPEG REL data model. The MPEG REL adopts a simple and extensible data model. In this data model, a license consists of one or more grants and issuers. And a grant contains four basic entities, including principal, right, resource, and condition. The roles of these elements of the MPEG REL data model will explained in detail in the following sections:

4.2.1 License



A license is a key top-level construct in the MPEG REL. It is a container of grants, issuers, and some other information. Conceptually, a license is a collection of grants issued by one or more issuers.

4.2.2 Grant

A grant is the element within the license that essentially conveys to a principal the right to use a resource, possibly subject to certain conditions.

4.2.3 Principal

MPEG REL provides the principal element that encapsulates the identification of a party to whom a right is granted. Each principal identifies exactly one party. In contrast, a set of principals, such as the universe of everyone, is not a principal. A

Principal is typically identified by using information unique to that entity, which often includes an associated authentication mechanism by which the Principal can prove its identity. For example, an X.509 certificate is one of an authentication proof.

To support identification, the following technologies are provided: 1) a principal must present multiple credentials, all of which must be simultaneously valid, to be authenticated; 2) a principal is a keyHolder, who is identified as possessing a secret key such as the private key of a public / private key pair. 3) Other identification methods that may be invented by others.

The following figure shows an example: the principal Jane is represented by using the keyHolder element.

```
<keyHolder licensePartId="Jane">
  <info>
    <dsig:KeyValue>
      <dsig:RSAKeyValue>
        <dsig:Modulus>SRagFBUrl...</dsig:Modulus>
        <dsig:Exponent>AQABAA==</dsig:Exponent>
      </dsig:RSAKeyValue>
    </dsig:KeyValue>
  </info>
</keyHolder>
```

Figure 4-3 Jane is identified as the holder of a particular key

4.2.4 Right

A right is the "verb" that a principal can be granted to exercise against some resources under certain conditions. Typically, a right specifies an action (or activity) or a class of actions that a principal may perform on or use the associated resource.

A right element within a grant encapsulates information about rights and provides a set of commonly used, specific rights, such as *play*, *print* and *adapt*, as well as notably rights relating to other rights, such as *issue*, *revoke* and *obtain*.

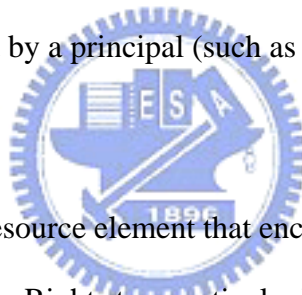
The following figure illustrates how the print right would appear in a license. In this figure, the prefix *mx:* indicates that the print right is defined in the MPEG media extension schema namespace that will be introduced in section 4.3.

```
<mx:print/>
```

Figure 4-4 The print right

4.2.5 Resource

A resource is the "object" to which a principal can be granted a right. A resource can be a digital work (such as an e-book, an audio or video file, or an image), a service (such as an email service, or B2B transaction service), or even a piece of information that can be owned by a principal (such as a name or an email address). A grant can also be a resource.



MPEG REL provides a resource element that encapsulates the information necessary to identify and assign Rights to a particular Resource. In particular, it provides a variety of pattern mechanisms to allow identification of a collection of resources with some common characteristics. Extensions to these mechanisms define Resources appropriate to specific business models. The following figure illustrates that an e-Book is identified as a digitalResource with a particular URI.

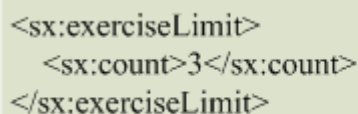
```
<digitalResource>  
  <nonSecureIndirect URI="http://www.contentguard.com/sampleBook.spd"/>  
</digitalResource>
```

Figure 4-5 An E-book is represented as a digitalResource with a URI

4.2.6 Condition

A condition specifies the terms, conditions and obligations under which rights can be exercised. A simple condition is a time interval within which a right can be exercised. A slightly complicated condition is to require the existence of a valid, prerequisite right that has been issued by some trusted entity. Using this mechanism, the eligibility to exercise one right can be dependent on the eligibility to exercise other rights.

MPEG REL defines a condition element that encapsulates the information necessary to determine whether the condition is met. Figure 4-7 presents constraint condition that identifies a right may be exercised only 3 times. The prefix *sx:* in this figure indicates that the elements (*exerciseLimit* and *count*) are defined in the MPEG standard extension schema namespace that will be introduced in section 4.3.



```
<sx:exerciseLimit>
  <sx:count>3</sx:count>
</sx:exerciseLimit>
```

Figure 4-6 A constraint condition

4.2.7 Issuer

The issuer element identifies a license issuer who digitally signs a license and provides additional details about its issuance. By signing the license, a license issuer authorizes the grants contained in a license and it can facilitate reliable establishment of trustworthiness of the license.

A license may have any number of issuers, or the element may be omitted entirely. However, no additional semantics is associated with the joint signing; it is as

if each had signed a copy of the License independently. No collective meaning is implied by several issuer elements appearing in a license.

The following figure illustrates how this issuer information is represented in the MPEG REL expression.

```
<issuer>
  <dsig:Signature>
    <dsig:SignedInfo>
      <dsig:CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
      <dsig:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
      <dsig:Reference>
        <dsig:Transforms>
          <dsig:Transform Algorithm="http://www.xml.org/schema/2001/11/xml2core#license"/>
        </dsig:Transforms>
        <dsig:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
        <dsig:DigestValue>PB4QbKOQC0941tTExbj1/Q==</dsig:DigestValue>
      </dsig:Reference>
    </dsig:SignedInfo>
    <dsig:SignatureValue>allDoedpL...</dsig:SignatureValue>
    <dsig:KeyInfo>
      <dsig:KeyValue>
        <dsig:RSAKeyValue>
          <dsig:Modulus>g8NRYMG30...</dsig:Modulus>
          <dsig:Exponent>AQABAA==</dsig:Exponent>
        </dsig:RSAKeyValue>
      </dsig:KeyValue>
    </dsig:KeyInfo>
  </dsig:Signature>
  <details>
    <timeOfIssue>2000-01-27T15:30:00</timeOfIssue>
  </details>
</issuer>
```

Figure 4-7 Issuer

4.3 Extensibility and Profiling

MPEG recognizes that different applications require different levels of complexity and flexibility in REL and that specific industries and user communities may need to modify the language to better meet their specific needs. To facilitate easy mapping of the REL to these industry-specific applications, MPEG has developed a process of extending and profiling of the language. [8]

The extension process, including developing new verbs and schematic elements, enables individuals to define new elements specific to their requirements to improve efficiency in a specific domain. The syntax of MPEG REL is defined using the XML Schema and Namespace Recommendations by W3C [9],[10], which enables the REL to offer a high degree of richness and flexibility in its extensibility. The following figure illustrates the MPEG REL architecture.

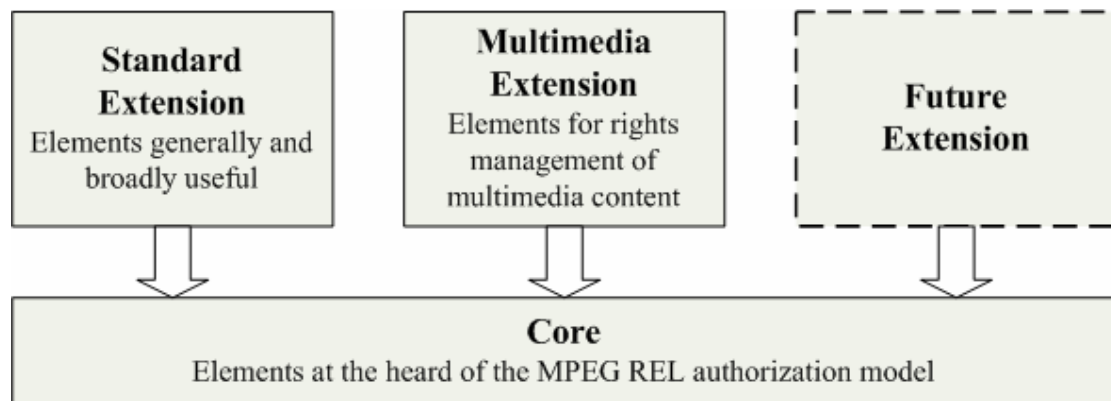


Figure 4-8 Extensibility structure.

These architectural parts: core, standard extension, multimedia extension, and their XML schemas are normative components of the overall MPEG REL specification.

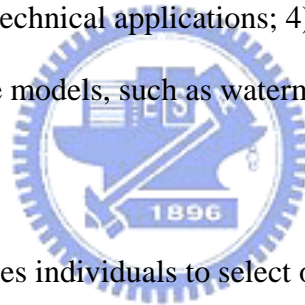
Core schema defines the general constructs which are essential and form the architectural basis for the language. The namespace for the MPEG REL Core is identified with the prefix “r:”. In some cases, this prefix is omitted.

Standard extension schema defines the concepts that are generally and broadly applicable to the DRM usage scenarios. It extends the condition type in the XrML core. The namespace for the MPEG REL standard extension is identified with the prefix “sx:”. Figure 4-6 is a simple example.

Multimedia extension schema extends the Core schema for concepts (e.g., rights, resources, and conditions) specifically related to the multimedia domain. As shown in Figure 4-4, the namespace for the MPEG REL multimedia extension is identified with the prefix “*mx:*”.

The extension process enables that the other parties can define their own (possibly domain-specific) extensions to the MPEG REL and its future extensions. This is accomplished by using the existing, standard XML Schema and XML Namespace mechanisms. More specifically, extensions to the MPEG REL can define:

- 1) principals appropriate to specific technical applications;
- 2) rights appropriate to using specific types of resource, such as play and print;
- 3) resources appropriate to specific business models and technical applications;
- 4) conditions appropriate to specific distribution and usage models, such as watermark, destination, and rendering application.



The profile process enables individuals to select only those language elements required to meet a specific application’s need. Many MPEG REL elements and attributes are optional, and their occurrences can be included or omitted in a profile to optimize payload of digital items and computation requirements of MPEG terminals. For a specific purpose or different devices, this involves the creation of a subset of the language. For instance, the full power of the MPEG REL, which is required by a PC, may not be available for a mobile phone. So a down-size version of this language, a new profile, can be created for this application.

Extension and Profile can be used concurrently to optimize the applicability of the MPEG REL to one specific application.

4.4 Authorization Model

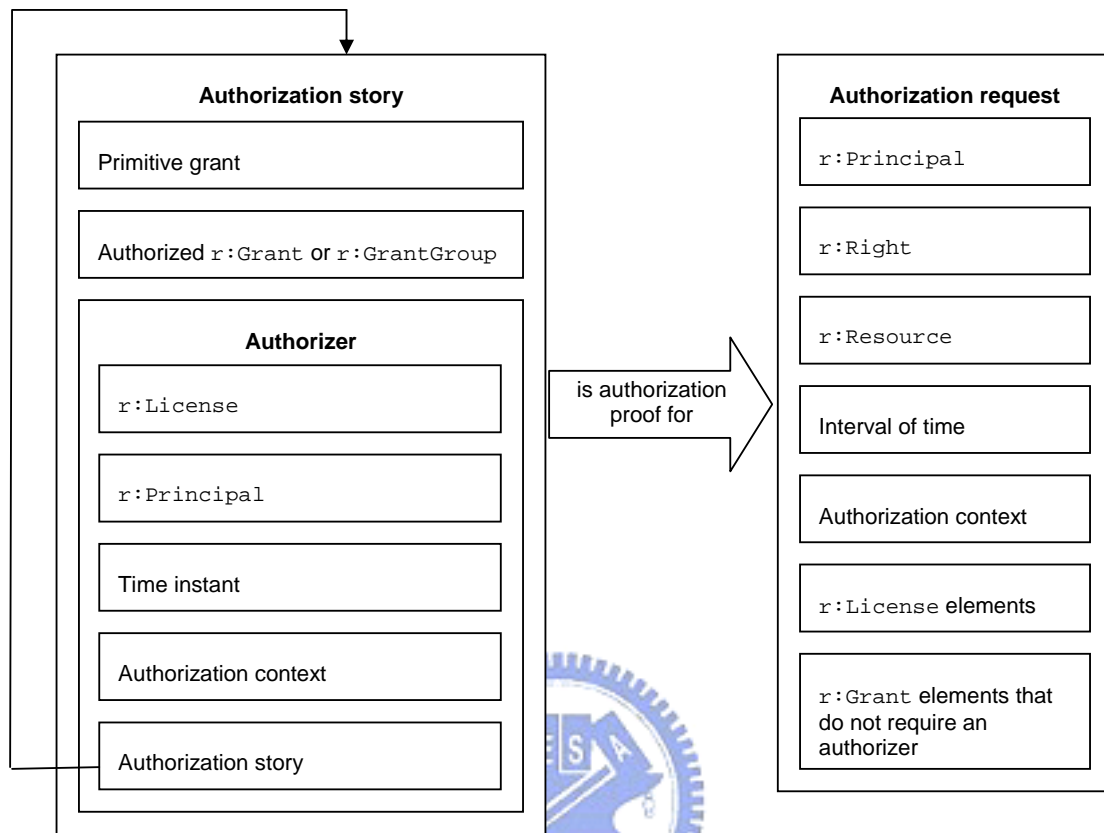


Figure 4-9 Authorization model [4]

The MPEG REL provides an authorization model, as shown in Figure 4-9, to determine if a principal has the right to perform an action on a resource according to the REL expressions in a set of licenses.

The authorization model makes use of an authorization request, an authorization story, authorization context, and an authorizer to create authorization proofs, which define the semantics of licenses for performing authorization requests. Figure 4-10 illustrates the relationship between the entities in the authorization model. An authorization request is permitted if there is one authorization story that provides an authorization proof for the request.

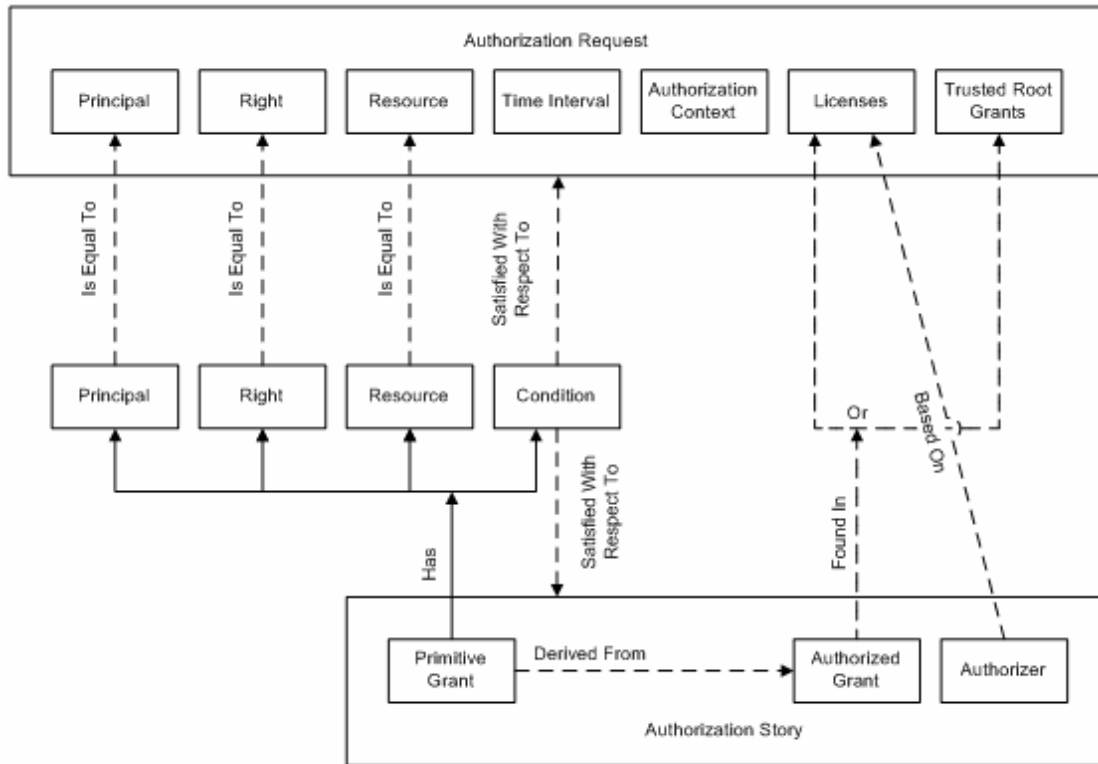


Figure 4-10 The relationships between the entities in the authorization model [11]



Authorization Request

An authorization request contains the information that conceptually represents the question "is it permitted for a given Principal to perform a given Right upon a given Resource during a given time interval based on a given authorization context, a given set of Licenses, and a given trust root?"

Authorization Story

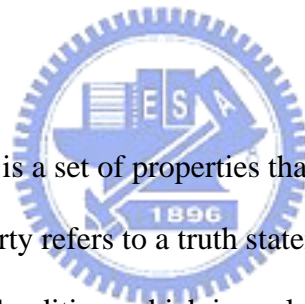
An authorization story is a structure created for the purpose of finding a grant to match the requested permission. An authorization story contains:

- A primitive grant containing no variables, derived from a grant that is authorized by an authorizer and is a part of the set of licenses provided in the authorization request.
- Optionally, an authorizer is a structure containing the information about issuance of the License.

Conceptually, an authorization story contains all the information necessary to state the following fact: “A given primitive grant (i.e., a grant with no variables) may be derived from a given grant authorized by a given authorizer, which identifies a principal from a license at a time instant based on an authorization context and supported by an authorization story.”

Authorization Context

An authorization context is a set of properties that are relevant to an authorization request. A property refers to a truth statement that is used in the evaluation of a Condition. A Condition, which is evaluated based on some authorization context properties, refers to those properties in its semantics. [11]



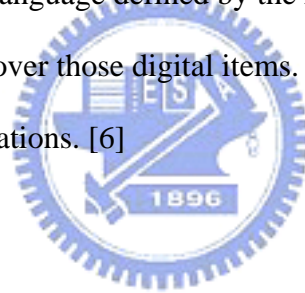
Authorization Proof

If all the relationships shown by the broken lines are valid in Figure 4-10, an authorization story is said to be an authorization proof for an authorization request. More specifically, the request is permitted if the following are true: 1) the primitive grant in the story is equal to the principal, right, and resource in the request and its condition is satisfied with the authorization context in the request; 2) the primitive grant must be derived from the grant; 3) the issuance of the grant in the first authorization story is recursively authorized via an additional authorization story in

the authorizer in the first authorization story; 4) the recursive authorization eventually terminates in one of the trust roots given in the authorization request. [6] When the request is permitted, there shall be at least one authorization story that supports an authorization proof for the authorization request.

4.5 The Relation with The Other standards in MPEG-21

The MPEG-21 REL [4] can declare rights and permissions using the terms as defined in the Rights Data Dictionary (RDD) [12]. The RDD specifies a dictionary of terms and provides a methodology for extending the dictionary to include new terms. The MPEG REL also can use the Digital Item Identification (DII) [13] to identify digital items described in the language defined by the Digital Item Declaration (DID) [14] and hence specify rights over those digital items. The MPEG REL can also be included in digital item declarations. [6]

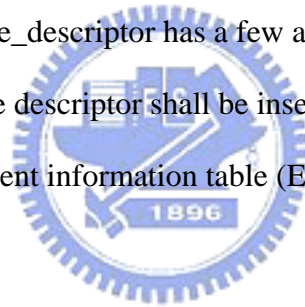


Chapter 5

Application Exampmples of the DRM system

In this chapter, we describe our implementations of DRM systems in which we use rights expression languages (REL) to implement an effective digital rights management (DRM) scheme on a DVB-T platform. In the implementations, we insert the REL license into the service information tables (SI) of DVB and it is broadcasted along with the content carried by the MPEG-2 transport bit stream. When a new TV program begins, a new License is broadcasted.

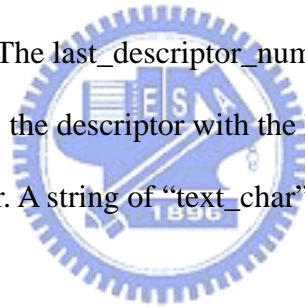
In order to insert the license into SI, we define a descriptor, `license_descriptor`, as show in Table 5-1. This `license_descriptor` has a few attributes and their syntax are defined in Table 5-1. Then, the descriptor shall be inserted into the corresponding `descriptor_loop` field of the event information table (EIT) as shown in Table 5-2.



| Syntax | No. of bits | Identifier |
|---|-------------|------------|
| <code>license_descriptor() {</code> | | |
| <code>descriptor_tag</code> | 8 | uimsbf |
| <code>reserved</code> | 4 | uimsbf |
| <code>descriptor_length</code> | 12 | uimsbf |
| <code>descriptor_number</code> | 8 | uimsbf |
| <code>last_descriptor_number</code> | 8 | uimsbf |
| <code>for (i=0 ; i<N ; i++) (</code> | | |
| <code>text_char</code> | 8 | uimsbf |
| <code>}</code> | | |
| <code>}</code> | | |

Table 5-1 License Descriptor

The descriptor_tag of license_descriptor is an 8-bit field. According to the specification of ETSI EN300 468 [2], the tag value from 0x80 to 0xFE could be defined by user. We define the descriptor tag value of the license_descriptor is 0x80. The “reserved” bits shall be set to 1. The descriptor_length is a 12-bit field specifying the total number of bytes of the data portion of this descriptor following the bytes defining the value of this field. Note the descriptor_length shall not exceed 4063 so that the entire descriptor has a maximum length of 4066 bytes. The descriptor_number gives the index of the descriptor. It is used to a license which is cannot be fitted into a single descriptor. The descriptor_number of the first license_descriptor of an associated set of license_descriptors shall be “0x00”. Within each segment of a license the descriptor_number shall increment by 1 with each additional license_descriptor. The last_descriptor_number specifies the index of the last license_descriptor (that is, the descriptor with the highest descriptor _number) of the associated set of descriptor. A string of “text_char” field specifies the text of a license.



Two DRM applications using this mechanism, broadcasting license within SI tables of MPEG-2 transport stream, are implemented. The first application is Digital TV Conditional Access (CA). We will describe its architecture and the demo system implementation in the next section. In the second application, we use an MPEG-21 REL Reference software to interpret an REL license for managing the right of using the contents. And we will show a “Validity Interval” condition case.

| Syntax | No. of Bits | Identifier |
|------------------------------|-------------|------------|
| event_information_section(){ | | |
| table_id | 8 | uimsbf |
| section_syntax_indicator | 1 | bslbf |
| reserved_future_use | 1 | bslbf |
| reserved | 2 | bslbf |
| section_length | 12 | uimsbf |
| service_id | 16 | uimsbf |
| reserved | 2 | bslbf |
| version_number | 5 | uimsbf |
| current_next_indicator | 1 | bslbf |
| section_number | 8 | uimsbf |
| last_section_number | 8 | uimsbf |
| transport_stream_id | 16 | uimsbf |
| original_network_id | 16 | uimsbf |
| segment_last_section_number | 8 | uimsbf |
| last_table_id | 8 | uimsbf |
| for(i=0;i<N;i++){ | | |
| event_id | 16 | uimsbf |
| start_time | 40 | bslbf |
| duration | 24 | uimsbf |
| running_status | 3 | uimsbf |
| free_CA_mode | 1 | bslbf |
| descriptors_loop_length | 12 | uimsbf |
| for(i=0;i<N;i++){ | | |
| descriptor() | | |
| } | | |
| } | | |
| CRC_32 | 32 | rpchof |
| } | | |

Table 5-2 Event Information Section [2]



5.1 Application One – Digital TV Conditional Access

In general, conditional access system of digital TV follows the classical DVB Condition Access System (CAS) mechanism as described in section 2.3. The Entitlement Control Messages (ECMs) and Entitlement Management Messages (EMMs) are broadcasted. They carry the information for decrypting the content.

In this application example, we present how to use REL license to implement another conditional access scheme. This implementation uses cryptography and key management machine to protect content from illegal access.

5.1.1 Cryptography

We use two security techniques including symmetric and asymmetric cryptography to encrypt content in this implementation.

5.1.1.1 Symmetric Cryptography

Symmetric cryptography, also called secret key cryptography (SKC) or private key cryptography, is the most popular of cryptography method. With symmetric cryptography, a single cryptographic key is used for both encryption and decryption. As shown in Figure 5-1.A, the sender uses a key to encrypt the plaintext and sends the cipher text to the receiver. The receiver must apply the same key to decrypt the message and recover the plaintext. Symmetric algorithms generally can be divided into stream ciphers and block ciphers. Stream ciphers encrypt a single bit (byte or computer word) of the message at a time and implement some form of feedback mechanism so that the key is constantly changing. The most widely used stream cipher is RC4 (Rivest Ciphers) algorithm. In Block ciphers a key and algorithm are applied to blocks of data rather than individual bits in a stream. Data Encryption Standard (DES) and Advanced Encryption Standard (AES) are the common block encryption algorithm.

Here, we will introduce the RC4 algorithm we used in this implementation. RC4 was developed by Ron Rivest in 1987. It is a stream cipher. This means it essentially functions as a random number generator whose output is combined with the plaintext by using XOR.

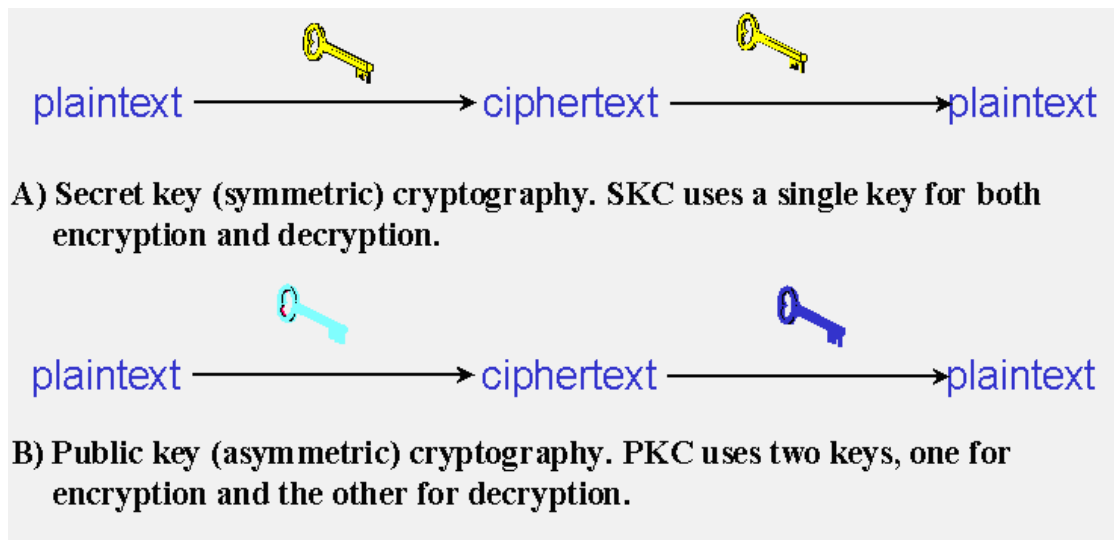


Figure 5-1 Cryptography [15]

RC4 has two stages in the procedure of generating random numbers. The first stage is the key-scheduling algorithm (KSA). It initializes a permutation (denoted “S” below) based on a variable length key. The algorithm is described below.

```

for i = 0 .. N-1
    S[i] = i;
j=0;
for i = 0 .. N-1
    j=j + S[i] + K[i % 1];
    Swap (S[i], S[j]);

```

After initialization, the second step is the pseudo-random generation algorithm (PRGA). It actually returns a random byte. Here is the algorithm.

```

i = j = 0;
i = i+ 1;
j= j + S[i];
Swap (S[i], S[j]);
return S[ S[i] + S[j] ];

```

The return value is XORed with the plaintext to produce the ciphertext, or the ciphertext to produce the plaintext.

5.1.1.2 Asymmetric Cryptography

Asymmetric cryptography, also called public key cryptography (PKC), was invented by Diffie and Hellman in 1976 [16]. The essential difference to symmetric cryptography is that this kind of algorithm uses two different keys for encryption and corresponding decryption. As shown in Figure 5-2.B, public key cryptography allows users to communicate securely without having to share a secret key by using a pair of cryptographic keys, designated as public key and private key, which are related mathematically. One key is used to encrypt the plaintext and the other key is used to decrypt the cipher text. Which key is applied first is not important. Who has the private key can share the related public key to others, but the private key can not be known by anyone.



RSA is the best known and most frequently used reversible asymmetric algorithm. It is named for its inventors Rivest, Shamir and Adelman [17]. The algorithm is described below.

1. Key Generation:

(a) Choose two prime number p and q and let $n=pq$.

(b) $\phi(n) = (p-1)(q-1)$.

The values of p , q , and $\phi(n)$ should be kept secret.

(c) Choose a random number E such that $\gcd(E, \phi(n)) = 1$.

(d) Compute an exponent D such that $ED \equiv 1 \pmod{\phi(n)}$.

The public key is the pair of values (n,E) and the private key is (n,D) .

n is known as the modulus.

E is known as the public exponent or encryption exponent.

D is known as the secret exponent or decryption exponent.

2. Encryption

Let P_i is the block of plaintext message and its length is smaller than $\phi(n)$.

The encrypted text C_i is calculated from the equation $C_i \equiv P_i^E \pmod{n}$.

3. Decryption

We calculate $P_i = C_i^D \pmod{n}$.

5.1.2 Key Management

This section presents a key management mechanism which is used within the conditional access scheme. The key management mainly uses a trans-encrypting operation. The entire scheme of this conditional access application is composed of a service provider, receivers, and a license server, as shown in Figure 5-2.

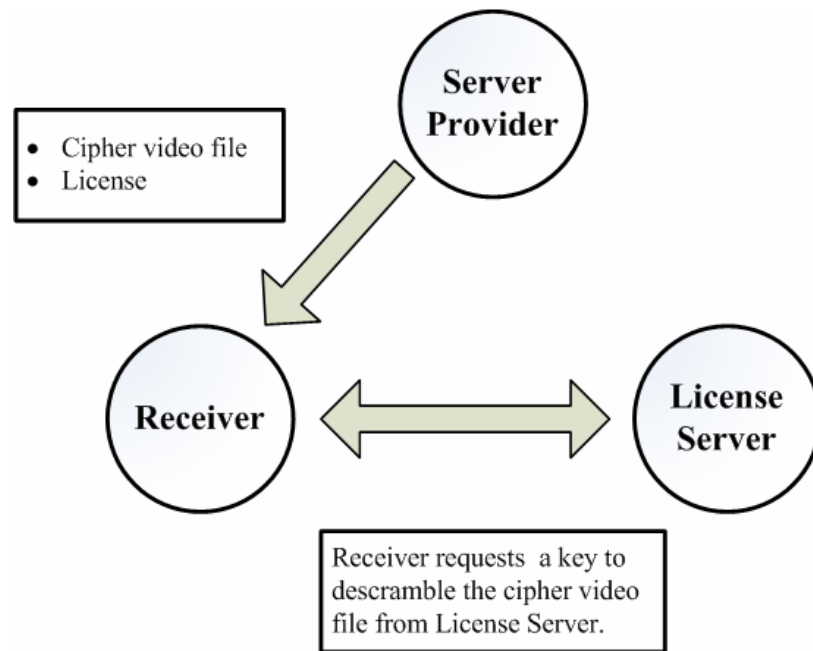


Figure 5-2 General idea of the conditional access scheme

Service provider broadcasts a scrambled TV program and its associated REL license to receiver. The scrambling key used to encrypt the TV program is encrypted by the license server's public key and put into the license. Here we use RC4 and RSA algorithms to scramble the program and encrypt the RC4 scrambling key, respectively. In addition to the encrypted scrambling key, this license also contains the license server information. Using the license information, the receiver send a request to the license

server for the scrambling key. This message includes both the encrypted RC4 key and receiver's name. The license server will check if the receiver has the right to watch the program or not. If the receiver has the right to watch the program, the license server does two things. It first decrypts the RC4 scrambling key by using its own private key. In order to protect the RC4, the license server encrypts the scrambling key by using the receiver's public key to ensure the receiver's identity.

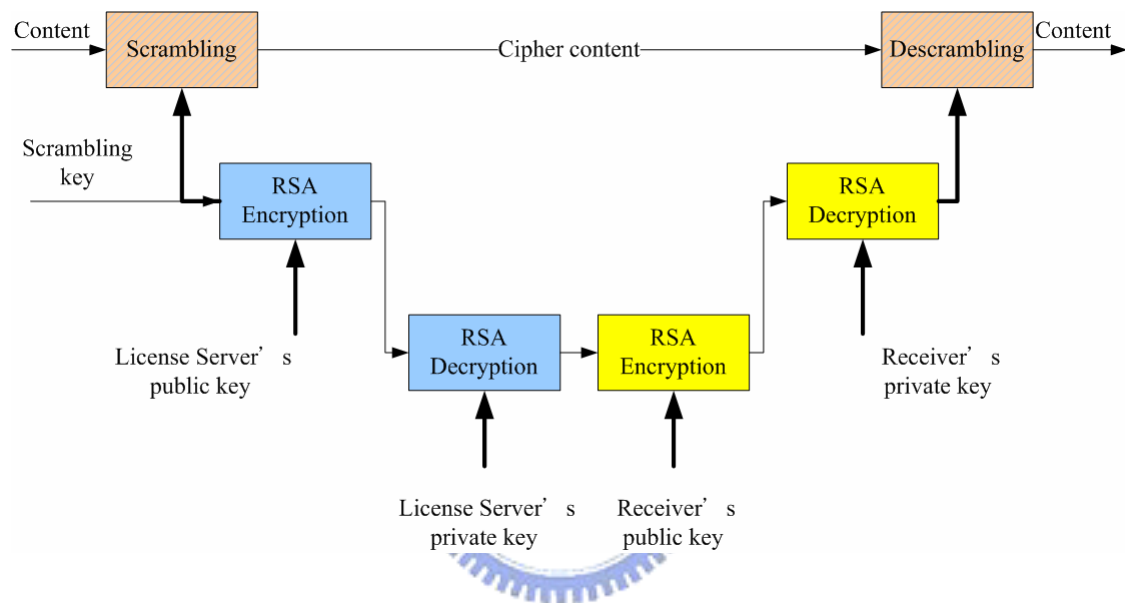


Figure 5-3 Key management procedure

Figure 5-3 shows how the keys are exchanged. According to our proposal, the content scrambling key is inserted into the license information carried by the content bit stream. Therefore, we can reduce the real-time synchronization requirement imposed on the communication between the service provider and the license server. Typically, the scrambling key is changed very frequently, for example, once per second.

5.1.3 Architecture Design

Figure 5-4 is the overall architecture of this conditional access scheme. There are mainly three parts: Service Provider, Receivers, and License Server. We now elaborate on their structures and operations.

5.1.3.1 Service Provider

The service provider part acts as a transmitter for transmitting the DTV broadcast and the related information. As shown in Figure 5-4, the service provider broadcasts a scrambled MPEG-2 video elementary stream and its associated REL license to receivers. The scrambling key is encrypted by the license server's public key and then is inserted into the REL license. The syntax of the license is based on the MEPEG-21 REL[4], Digital Item Declaration (DID) [14], and Digital Item Identification (DII) [13]. In addition to the encrypted scrambling key, the license also contains the license server's information. In the service provider part, this license is inserted into the service information table (SI). The multiplexer takes in the encrypted MPEG -2 stream and converts it into a transport stream with the SI. Then, the consumers can use a DTV Card to receive the broadcast programs. The functionalities of each component are described below.

Scrambler: The Scrambler encrypts the data. Typically a cryptography is adopted for its simple operation. The algorithm used to encrypt the data in this application is RC4 stream encryption algorithm. The key used to encrypt the content is called the scrambling key.

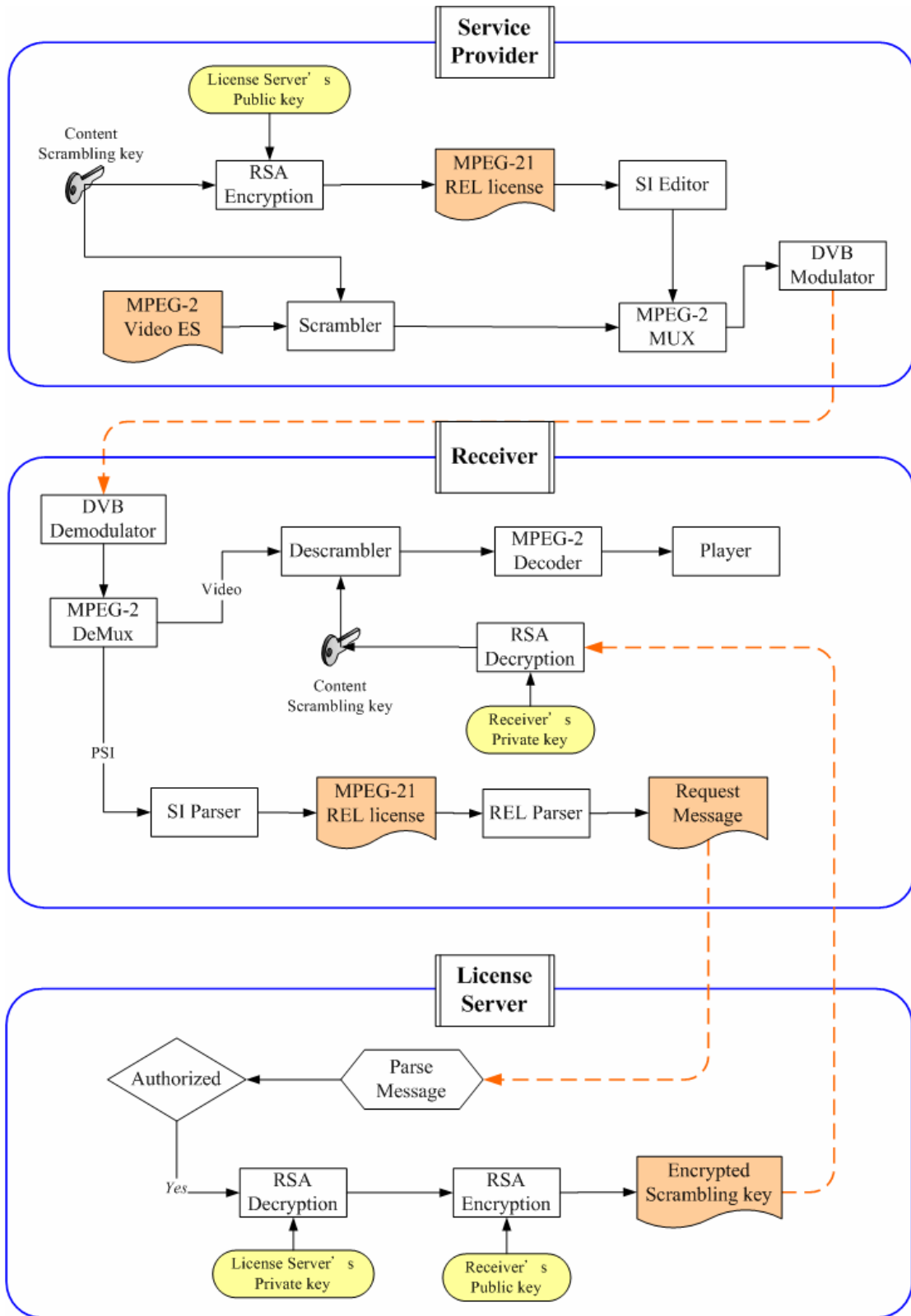
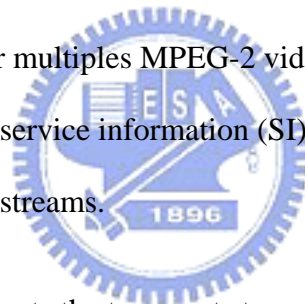


Figure 5-4 Architecture of the conditional access scheme on DVB-T system

RSA Encryption: It is responsible to encrypt the scrambling key by using an asymmetric cryptography structure. The input, scrambling key, is encrypted by the license server's public key. When the user contacts the license server, the scrambling key is encrypted by the user's public key. The algorithm used in this application is the RSA asymmetric encryption algorithm.

SI Editor: The SI editor is responsible to edit the EIT tables. First, a REL license is segmented into license_descriptors according the syntax in Table 5-1. Then, the descriptor is inserted into EIT as show in Table 5-2. Finally, in order to be multiplexed into a transport stream later, the EIT is packetized and the data is stored in the transport packets.

Multiplexer: Multiplexer multiples MPEG-2 video bit stream with program service information (PSI) and service information (SI) and produces the MPEG-2 Transport (ISO/IEC 13818-1) streams.



DVB Modulator: It converts the transport stream to an analog signal so that we can broadcast it.

5.1.3.2 Receiver

The receiver part is responsible to receive and decode a digital video broadcast program transmitted by the service provider and render it by a display device such as a television or a monitor.

The architecture of the receiver is shown in Figure 5-4. Before descramble the MPEG-2 video stream, the license shall be parsed for generating a message to request for the content scrambling key. The request message includes the information

extracted from the received license. The message contains the receiver's name for identification, the identifier of the MPEG-2 digital content, the right that the consumer wants to exercise, and the content scrambling key encrypted by the license server's public key. For security reason, the license server will send the content scrambling key encrypted by the receiver's public key if the consumer was authorized to consume the digital content by the license server. So, the receiver needs to decrypt the scrambling key by the receiver's private key after receiving the key from the license server. Finally, the consumer can descramble the TV program for decoding and displaying.

The functionalities of each component are described below.

Demultiplexer: It takes in the MPEG-2 transport stream from the server provider and demultiplexes the bit stream based on packet IDs (PIDs).

SI Parser: It retrieves the SI tables from an MPEG-2 transport stream and parses the license_descriptor to retrieve the MPEG-21 REL license.

REL Parser: It parses the MPEG-21 REL license and extracts the license server's information and the encrypted scrambling key from the license to make the request message.

RSA Decryption: It is responsible to use the receiver's private key to decrypt the scrambling key sent from the license server. The algorithm used in this application is the RSA asymmetric encryption algorithm.

Descrambler: Using the scrambling key to decrypt the TV program. The algorithm used to decrypt the cipher text in this application is RC4 stream decryption algorithm.

5.1.3.3 License Server

License Server is a third party to authorize the receiver to access the scrambling key by using a trans-encrypting operation.

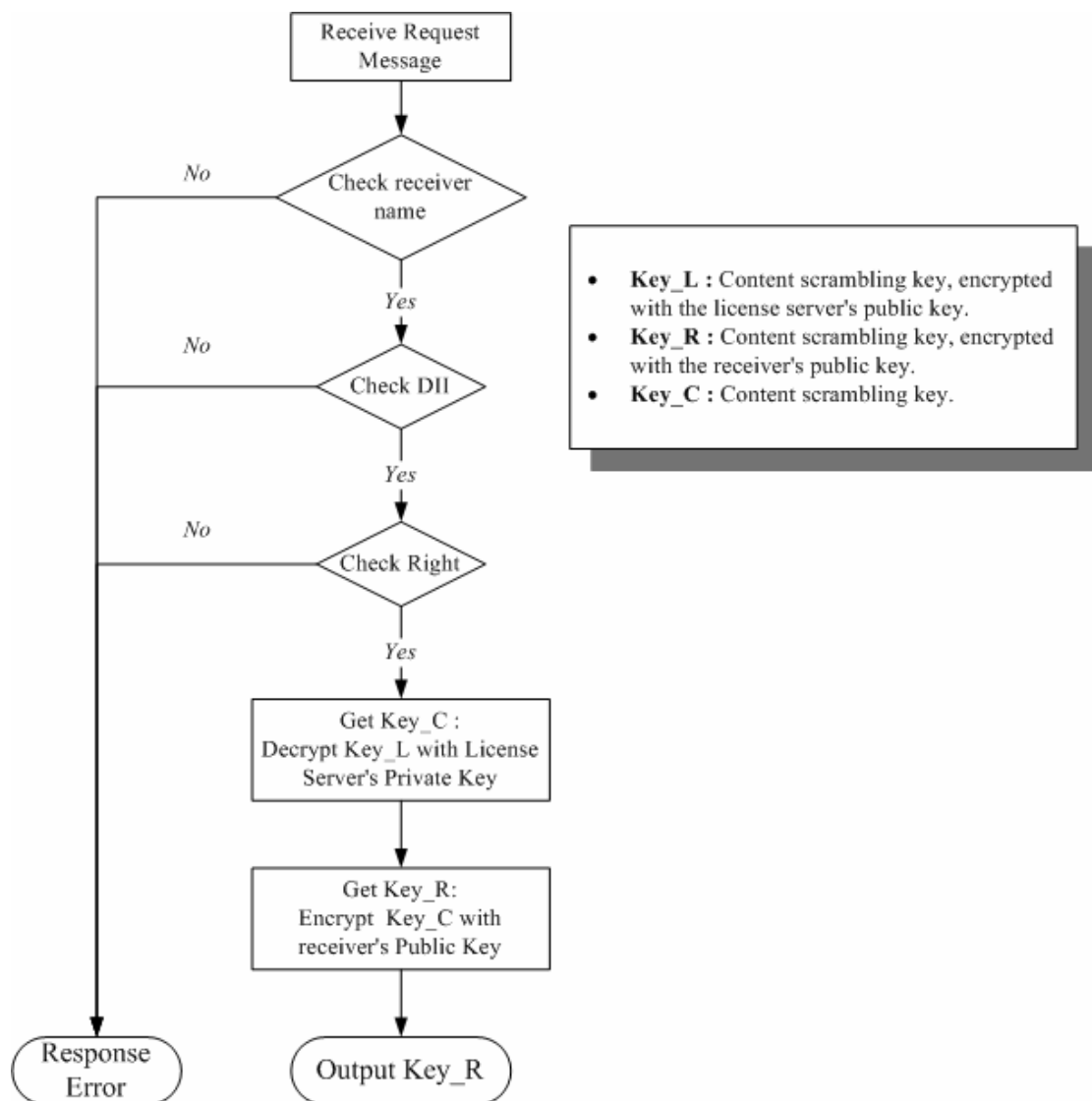


Figure 5-5 Flow chart of the license server

The flow chart of the license server operation is shown in Figure 5-5. The license server will check the right of the receiver as soon as it receives the request message. If the receiver has the right to exercise this TV program, the license server will use its private key to decrypt the scrambling key contained in the request message. After decryption, the license server also needs to encrypt the scrambling key by the receiver's public key for protection. At this step, the encryption can protect the key from being used by illegal receiver.

5.1.4 An Implementation of the Condition Access Scheme

This implementation uses the Twinhan DVB-T PCI Card for the Receiver part to receive and demultiplex the DVB-T signal. The SDK of the Twinhan DVB-T PCI Card is described in chapter 3.

To match the SDK of the Twinhan DVB-T PCI Card, the receiver part of this conditional access scheme and the next application example are both written in C++. And because of hardware restrictions of the DVB-T PCI Card, we can not perform the real-time descramble for playing back when the bit stream is received in real-time. So, we will download the TV program stream at the receiver for performing offline conditional access.

Figure 5-6 shows the filter graph of the DVB-T PCI Card in this implementation. We add two filters, the MPEG-2 Sections and Tables filter and the Sample Grabber filter, into the filter graph of the DVB-T PCI Card.

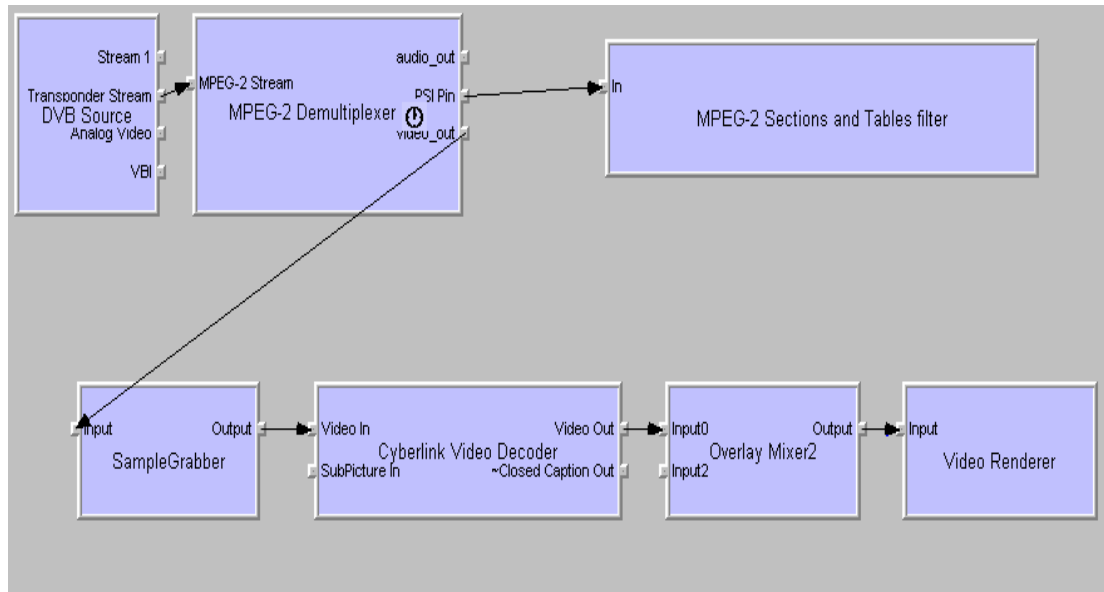


Figure 5-6 Filter graph of the DVB-T PCI Card in the conditional access scheme

The MPEG-2 Sections and Tables filter, provided by Microsoft® DirectX®, retrieves the Program Specific Information (PSI) tables from an MPEG-2 transport stream. We can use this filter to obtain any PSI data, such as ATSC Program and System Information Protocol (PSIP) tables, DVB service information (SI), conditional access tables (CATs), DSM-CC messages, or private table data [18]. The MPEG-2 Sections and Tables filter is connected to an output pin of the MPEG-2 Demultiplexer. It acts as the SI parser as shown in Figure 5-4.

The Sample Grab filter, also provided by Microsoft® DirectX®, provides a way to retrieve samples as they pass through the filter graph. It is a transform filter with one input pin and one output pin. It passes the downstream unaltered. The filter has no preferred media type. This means it can be connected to almost any filter in the graph, but then we do not have control on the type of data that it outputs. Because the hardware restriction of the DVB-T PCI Card, we could not directly add the descrambling process in the filter graph of the SDK for the scrambled video stream.

Therefore, we use the filter to retrieve the scrambled video stream and save it in a file. Then, the file will be descrambled in a separate process as shown in Figure 5-8.

Figure 5-7 shows the server provider part and the DVB-T PCI card part of the receiver. We use the SDK of the Twinhan DVB-T PCI card to receive and demultiplex the DVB signal, parse the SI tables, and download the TV program to save in a video file.

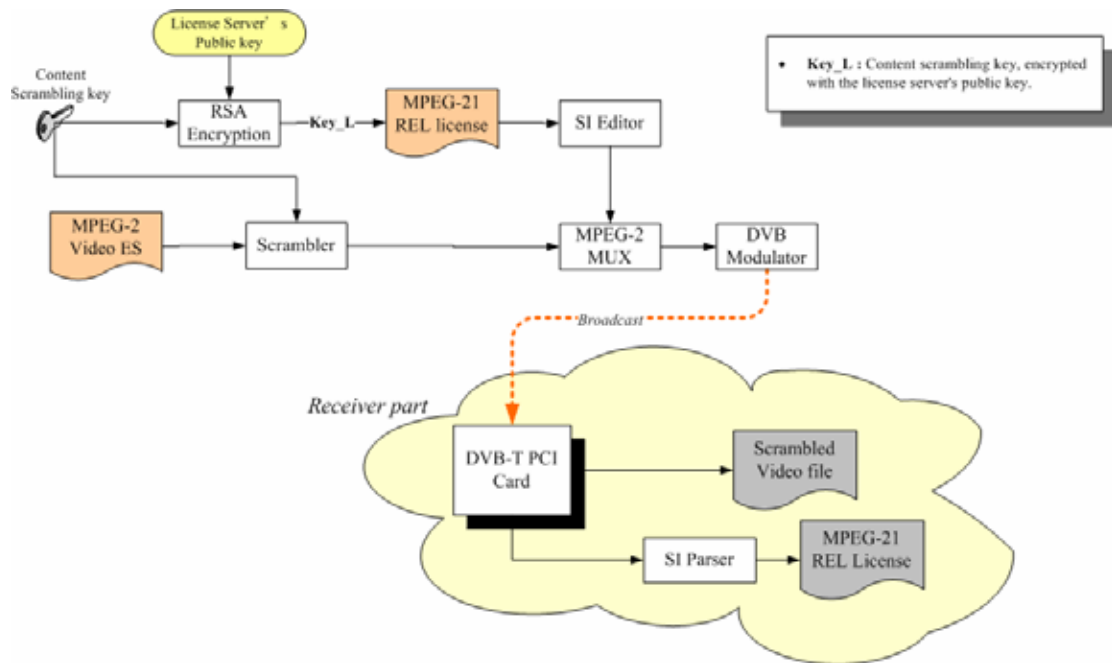


Figure 5-7 The server provider part and the DVB-T PCI card part of the receiver

Figure 5-8 shows the dataflow after retrieving the REL license and the downloaded video file. We use an Internet Information Services (IIS) Web server hosted on the local host as the license server for this demonstration. The receiver sends a request message to request an XML data file from the license server, a local web server, using HTTP. The XML data consists of the scrambling key information

that is stored as an XML file on the server. The scrambling key is encrypted by the receiver's public key. The license server fulfils the request through an ASP page, which is accessible from the localhost virtual directory.

In our implementation, we use a library called Kivco^{RSA} as the RSA Encryption and the RSA Decryption components. It is a small library of encryption routines that communicate with the ASP using the ATL COM standard. It provides many implementations of cipher algorithm, such as AES, DES, RSA, and many others. And the library provides both stream ciphers and block ciphers [19].

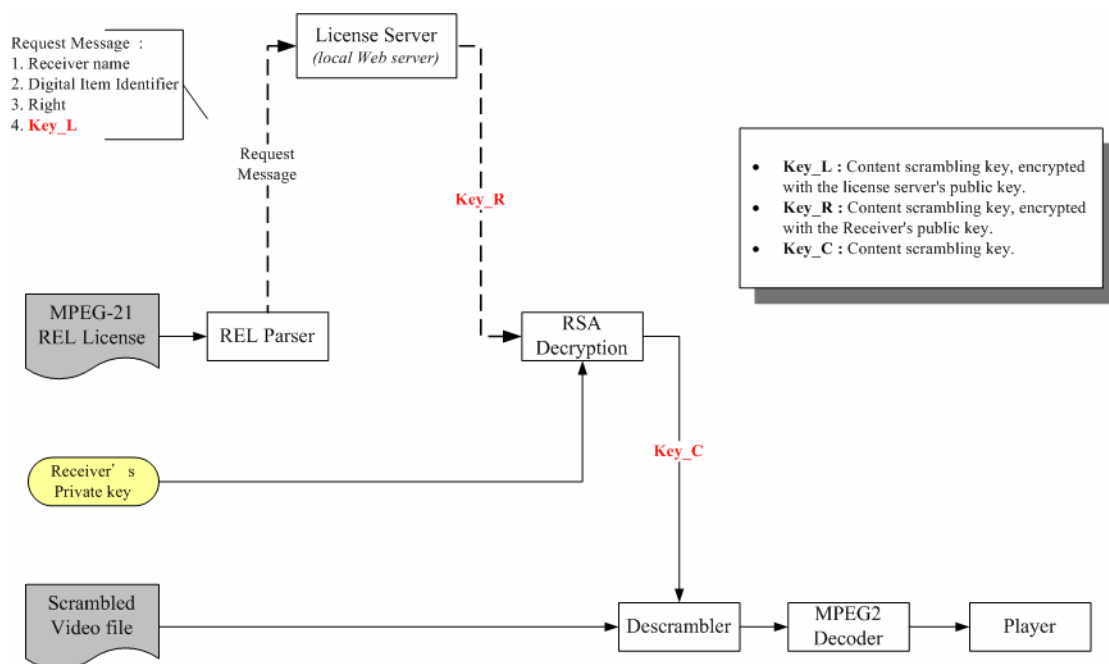


Figure 5-8 Dataflow after retrieving the REL license and the downloaded video file

5.1.5 Conditional Access Demonstration System

In this section, we will describe this implementation of the condition access scheme described in the previous sections. The license attached below (XML codes) shows that the license server grants the subscribers to watch the TV program. It also contains a description of the TV program and the scrambling key. This license and the video file are both broadcasted and received by the users. The license size is 2934 bytes.



```

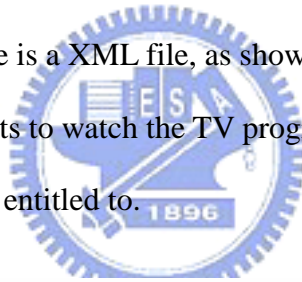
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited by Ying -->
<DIDL xmlns="urn:mpeg:mpeg21:2002:02-DIDL-NS" xmlns:dii="urn:mpeg:mpeg21:2002:01-DII-NS" xmlns:dsig="http://www.w3.org/2000/09/xmldsig#"
xmlns:enc="http://www.w3.org/2001/04/xmlenc#" xmlns:mx="urn:mpeg:mpeg21:2003:01-REL-MX-NS" xmlns:r="urn:mpeg:mpeg21:2003:01-REL-R-NS"
xmlns:sx="urn:mpeg:mpeg21:2003:01-REL-SX-NS" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Container>
    <Descriptor id="SharingLicense">
      <Statement mimeType="text/xml">
        <r:license>
          <r:grant>
            <!-- Identification (by name and public key) of the license server that is given the right to issue new domain licenses -->
            <r:keyHolder>
              <r:info>
                <dsig:KeyName>http://localhost/sxh/LicenseServer.asp</dsig:KeyName>
                <dsig:KeyValue>
                  <dsig:RSAKeyValue>
                    <dsig:Modulus>B40595355D8CD5E45FDBF40DD6F3EA43A0B3CD3BBDC824D914553655EAF2220107D
                    E66044C4F356298F1326DD25944BD</dsig:Modulus>
                    <dsig:Exponent>AAAB</dsig:Exponent>
                  </dsig:RSAKeyValue>
                </dsig:KeyValue>
              </r:info>
            </r:keyHolder>
            <r:issue/>
            <!-- Type of license to be issued to end-users -->
            <r:grantGroup>
              <r:grant>
                <mx:play/>
                <mx:diReference>
                  <mx:identifier>urn:uuid:14324da6-7d37-49c8-9656-05acadfd2b9c</mx:identifier>
                </mx:diReference>
              </r:grant>
            </r:grantGroup>
          </r:grant>
          <!-- Content scrambling key, encrypted with the license server's public key -->
          <r:otherInfo>
            <enc:EncryptedKey>
              <enc:ReferenceList>
                <enc:DataReference URI="enc_test.es"/>
              </enc:ReferenceList>
              <dsig:KeyInfo>
                <dsig:KeyName>http://localhost/sxh/LicenseServer.asp</dsig:KeyName>
              </dsig:KeyInfo>
              <enc:CipherData>
                <enc:CipherValue>1510b3de9d46780f34ac8bbdd</enc:CipherValue>
              </enc:CipherData>
            </enc:EncryptedKey>
          </r:otherInfo>
        </r:license>
      </Statement>
    </Descriptor>
    <Container id="ContentContainer">
      <Item id="ContentItem">
        <!-- Randomly generated unique item identifier-->
        <Descriptor>
          <Statement mimeType="text/xml">
            <dii:Identifier>urn:uuid:14324da6-7d37-49c8-9656-05acadfd2b9c</dii:Identifier>
          </Statement>
        </Descriptor>
        <!-- Title of the digital item-->
        <Descriptor id="ItemTitle">
          <Statement mimeType="text/plain">Cartoon</Statement>
        </Descriptor>
        <!-- Textual description of the digital item-->
        <Descriptor id="ItemDescription">
          <Statement mimeType="text/plain">This is a Cartoon.</Statement>
        </Descriptor>
        <Component>
          <!-- Reference to the MP2 video resource video.es, stored locally -->
          <Resource mimeType="video/mp2" ref="vedio.es"/>
        </Component>
      </Item>
    </Container>
  </Container>
</DIDL>

```

The attributes and parameters of the license_descriptor in the license above are described below.

```
license_descriptor() {  
    descriptor_tag    = 0x80  
    reversed          = 0xF  
    descriptor_length = 0xB78  
    descriptor_number = 0  
    last_descriptor_number = 0  
    text_char = {MPEG-21 REL license}  
}
```

In the license server, there is a XML file, as shown below, to record the subscribers, who have the rights to watch the TV programs, and the types of license permission the subscribers are entitled to.



```
<?xml version="1.0"?>  
<Database>  
  <User>  
    <name>Alice</name>  
    <PublicKey>65537</PublicKey>  
    <N>128356948827059</N>  
    <grant>  
      <right>play</right>  
      <diReference>  
        <identifier>urn:uuid:14324da6-7d37-49c8-9656-05acadfd2b9c</identifier>  
      </diReference>  
    </grant>  
  </User>  
  <User>  
    <name>Jane</name>  
    <right>play</right>  
    <PublicKey>65537</PublicKey>  
    <N>159838694882989</N>  
    <grant>  
      <right>play</right>  
      <diReference>  
        <identifier>urn:uuid:14324da6-7d37-49c8-9656-05acadfd2b9c</identifier>  
      </diReference>  
    </grant>  
  </User>  
</Database>
```

The following demo shows two different cases. The first case is that the receiver is one of the valid subscribers of the license server. Figure 5-9 is the console window shows the messages in this exercise. Figure 5-10 is the picture screenshot during the playing back.

```
C:\WINDOWS\System32\cmd.exe

D:\Thesis\Demo\App_1>UserSide license.xml Alice

License Server : http://localhost/sxh/LicenseServer.asp
Digital Item Identifier: urn:uuid:14324da6-7d37-49c8-9656-05acadfd2b9c
Right = play
Encrypted Key = 1510b3de9d46780f34ac8bbdd
Cipher Video File = video.es

>>> Conntet to the license server ...

Response Message ....
Response Encrypted Key : 16c2c86fffe47dd052e997e

After RSA Decryption,
the clear scrambling Key : 1234
The video file was opened

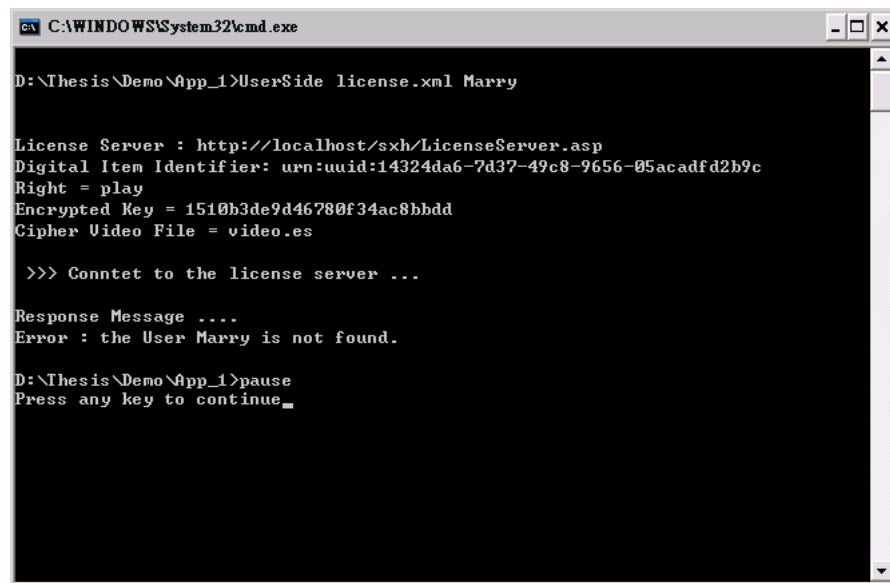
D:\Thesis\Demo\App_1>pause
Press any key to continue.
```

Figure 5-9 The console window of the first case



Figure 5-10 Screenshot

The second case is that the receiver who does not have the right to watch the broadcast TV program. Figure 5-11 is the console window shows the message in this process. The license server responds an error message; thus, the user can not watch the TV program.



```
C:\WINDOWS\System32\cmd.exe

D:\Thesis\Demo\App_1>UserSide license.xml Marry

License Server : http://localhost/sxh/LicenseServer.asp
Digital Item Identifier: urn:uuid:14324da6-7d37-49c8-9656-05acafd2b9c
Right = play
Encrypted Key = 1510b3de9d46780f34ac8bbdd
Cipher Video File = video.es

>>> Conntet to the license server ...

Response Message ....
Error : the User Marry is not found.

D:\Thesis\Demo\App_1>pause
Press any key to continue_
```

Figure 5-11 The console window of the second case

5.2 Application 2 –Validity Interval Condition

When the decoded DVB content is recorded at a DVB receiver, it can be re-played by any one. DVB Conditional Access (CA) provides addressability on broadcasting network by using the smart card, but it can not provide a further protection and manage the content usage policies after the DVB content is recorded or download at the DVB receiver. Therefore, we need a digital rights management (DRM) component to provide content providers a way to protect their products from unauthorized copying and other illegal uses.

In this application, we like to show that we can use a MPEG-21 REL license as the bridging between the DVB Conditional Access system and a DRM system. By

broadcasting the license and the DVB program stream together, the content provider or service provider can use the license to control the receiver right to use the digital content on trusted devices.

In the application, we assume the conditional access follows the classical DVB Condition Access System (CAS) mechanism. For the subscriber who has the right to access the DVB programs, we use the MPEG-21 REL license to control their right to use the programs. In the thesis, we will demo a case – Validity Interval Condition.

5.2.1 MPEG-21 REL Reference Software

After the subscriber receives the DVB program and the MPEG-21 REL license, we use an REL Reference software to interpret REL license. The REL Reference software is a simple REL interpreter developed by Content Guard [20]. It is based on the specification and schemas released as MPEG-21 REL [4]. It now supports the “play” right and the “validityInterval” and “exerciseLimit” conditions.

The implementation contains three components: RELicAuthzDriverGUI.exe, RELicAuthzDriver.dll, and RELicAuthz.dll. “RELicAuthz” is the key modules for the MPEG-21 REL authorization application. “RELicAuthzDriver” is the reference driver for MPEG-21 REL, which makes use of the reference interpreter. RELicAuthzDriverGUI.exe is the user interface for the application. It is this component that each of the Window PC and Pocket PC versions has its own version.

The implementation follows the dataflow shown in the following diagram:

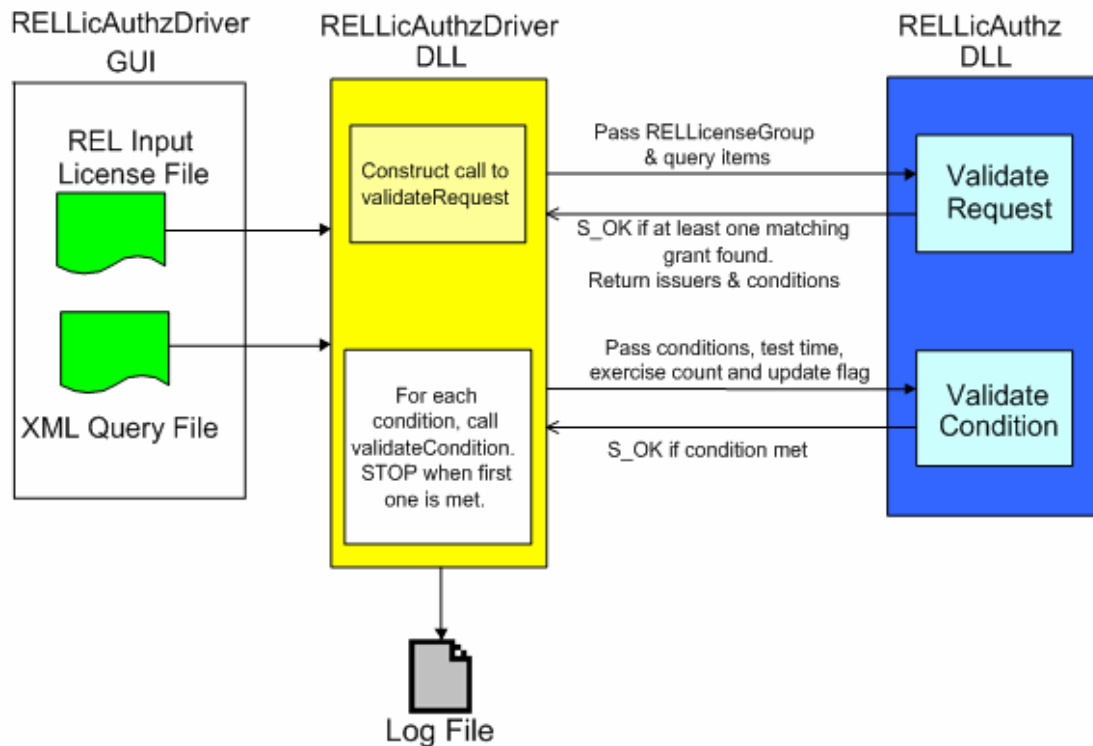


Figure 5-12 Dataflow of the reference software [20]

The two main modules in RELlicAuthz.dll provided in this implementation are validateRequest and validateCondition. The “validateRequest” module takes in an REL license file, checks to see if there are any matching grants against the supplied query items (keyHolder, right and resource). It returns vectors of conditions for any conditions found in the matching grants, and vectors of issuers for the issuers of the matching grants. The “validateCondition” module checks to see if any of the conditions is met. This implementation only supports the “validityInterval” and the “exerciseLimit” conditions. it takes in: 1) an input string containing either of these two conditions or an “allConditions” element containing either of the conditions; 2) a string containing an ISO date time to check; 3) an integer indicating the intended usage count; 4) and a Boolean variable indicating whether to update the state of the usage count.

When running the program, it displays a dialog box for selecting a MPEG-21 REL license file and a query file. A query file contains what license content is querying. It contains five things: the RSA modulus and exponent for a keyHolder principal, a right to be exercised, a resource URI, a time stamp to check against the validityInterval (this time stamp represents the expected exercise time), and a count indicating the intended usage count for the right. The query question asked would be:

Can the <principal> exercise the <right> <count> times on the <resource> at <time>?

This query file is not in the REL syntax. The query file below shows a sample example that follows the XML syntax.

```
<?xml version="1.0"?>
<query xmlns="http://www.contentguard.com/2002/exampleQueryNS">
  <goalToCheck>
    <principalToCheck
      rsaModulus="Fa7wo6NYfmvGqy4ACSWcNmuQfbejSZx7aCibIgkYswUeTCrmS0h27GJrA15SS7TYZzSfa
        S0xR9lZdUEF0ThO4w==" rsaExponent="AQABAA==" />
    <rightToCheck>
      <cx:print xmlns:mx="urn:mpeg:mpeg21:2002:01-REL-NS" />
    </rightToCheck>
    <resourceToCheck uri="http://www.contentguard.com/sampleBook.spd" />
  </goalToCheck>
  <simulationExerciseEnvironment>
    <time>2001-12-23T01:30:00</time>
    <exerciseCountToCheck update="true">2</exerciseCountToCheck>
  </simulationExerciseEnvironment>
</query>
```

Figure 5-13 shows the control flow of the reference software. The program execution flow is described as follows [20].

1. The user selects a REL license file name and a query file name.
2. RELlicAuthzCEGUI calls the relAuthorizeRight function by passing in the license file name and the query file name.
3. relAuthorizeRight loads the selected REL license file and the query file.

4. relAuthorizeRight calls validateRequest by passing a licenseGroup, user, right, resource to it.
5. validateRequest checks for unsupported items.
6. If no unsupported items are found, validateRequest loops through all the grants in all licenses. If at least one matching grant is found, it will return a positive result. Any conditions found are also returned.
7. If the call to validateRequest failed, the function RELAuthorizeRight exits.
8. If the call to validateRequest is successful, validateCondition will be called.
9. validateCondition loops through all the conditions to find at least one that is satisfied.
10. If we meet at least one condition, it means that the specified user is allowed to exercise the right on the given resource.
11. The user is informed of the success/failure of the authorization.
12. The user receives further information from the log file 'AuthorizationSession.log' at the directory which contains the application. (if 'Log Authorization Session' option is selected).

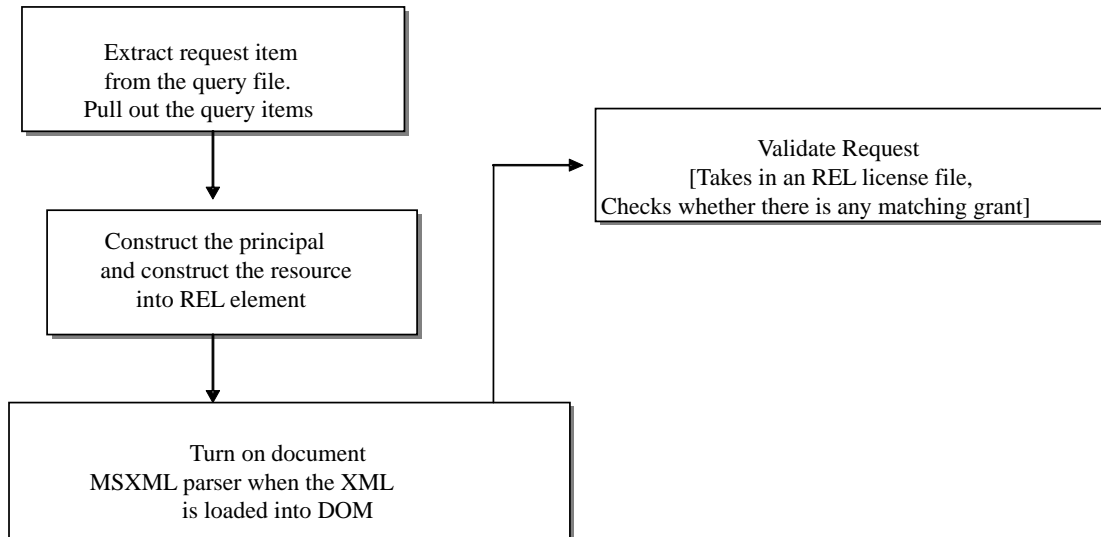
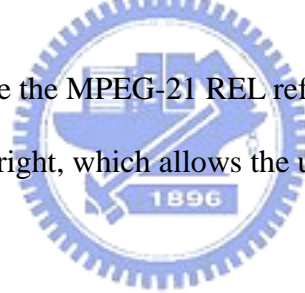


Figure 5-13 The control flow of the reference software [21]

5.2.2 Implementation of Validity Interval Condition

In this application, we use the MPEG-21 REL reference software at the terminal side to authorize the viewer's right, which allows the user to use the recording file according to the REL license.



At the server provider side, shown in Figure 5-14, the License of MPEG-2 video content is placed into the License Descriptors of Event Information Table, and the video file is encoded in an MPEG-2 transport stream. The Event Information Table is inserted into the MPEG-2 transport stream. Then the service provider broadcasts the MPEG-2 transport stream to the receiver. Assuming the conditional access follows the classical DVB Condition Access System (CAS) mechanism and the viewers discussed below have the right to access the program. When recording a DVB program, the content is saved in the MPEG-2 file format and the viewer receives also a MPEG-21 REL license simultaneously. In order to facilitate manipulation, the license is not stored within the MPEG-2 video file, it is stored in a separate file. As shown in Figure

5-14, we add a “Sample Grab Filter” between “MPEG-2 de-multiplexer” and “MPEG-2 Decoder” in Twinhan DVB-T PCI Card SDK. We use the filter to record the TV program in the MPEG-2 file format.

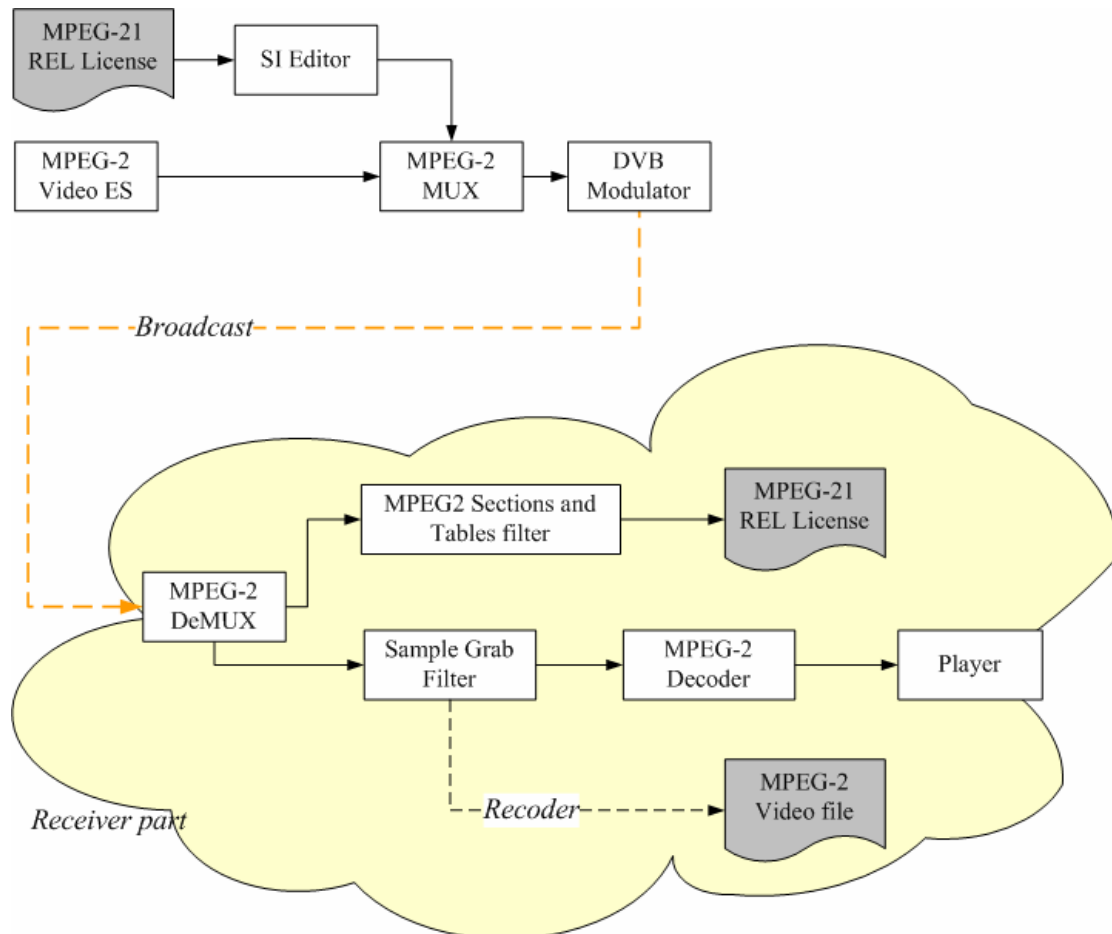


Figure 5-14 The server provider and the receiver in Application 2

After recoding, any one who wants to use the recoding file will need an REL tool to authorize the right according to the license. This implementation uses the MPEG-21 REL reference software described in section 5.2.1 as the REL tool for the purpose of interpreting and checking REL licenses. The Figure 5-15 below shows the control flow of rendering and authorization process in this application. At the control point, the MPEG-2 video file is decoded or not is decided by the authorization result

from the MPEG-21 REL reference software. If one of the grants in the license matches and at least one conditional is met with the user query, the authorization result is “Yes” and the user has the right to play the video file. Otherwise, the result is “No” and the file can not be rendered.

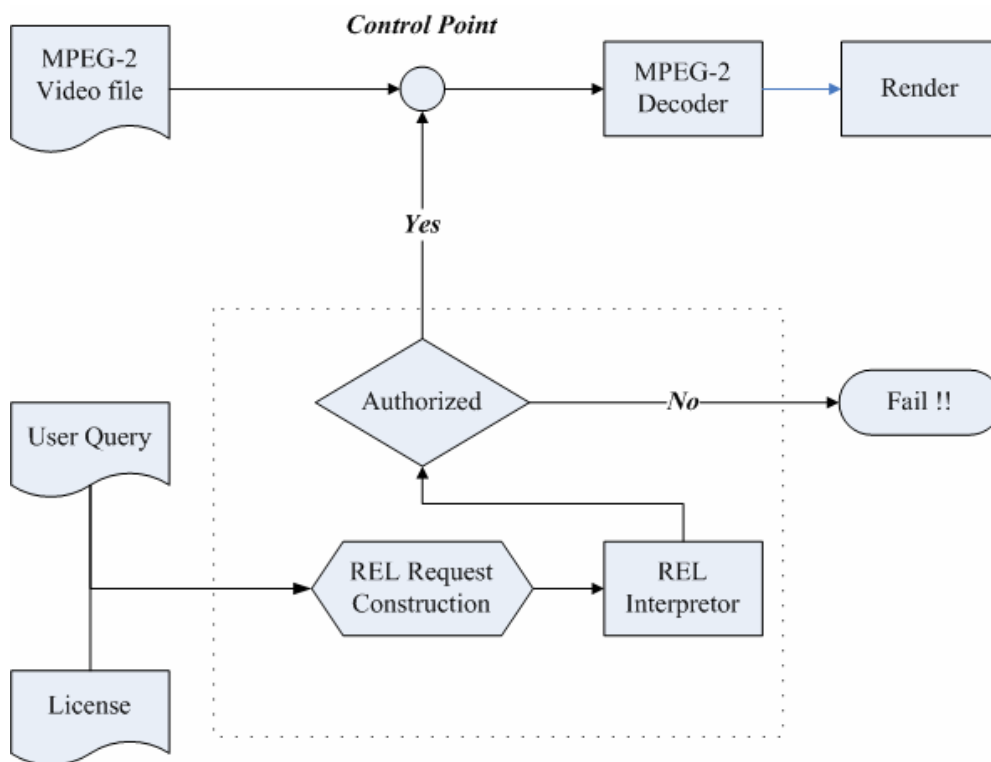


Figure 5-15 Control flow of rendering and authorization process

5.2.3 Demo of Validity Interval Condition Scenario

In this section, we will demonstrate a validity interval condition case. The license below shows that any one can play the video file before 2002/12/24 23:59:59. The license and the video file are both broadcasted to the receivers.

```

<?xml version="1.0" encoding="UTF-8"?>

<licenseGroup xmlns="urn:mpeg:mpeg21:2003:01-REL-R-NS" xmlns:sx="urn:mpeg:mpeg21:2003:01-REL-SX-NS" xmlns:dsig="http://
www.w3.org/2000/09/xmldsig#" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:mx="urn:mpeg:mpeg21:2003:01-REL-
MX-NS" xsi:schemaLocation="urn:mpeg:mpeg21:2003:01-REL-MX-NS ../schemas/rel-mx.xsd">
  <license licenseId="Example1">
    <!-- Anyone can play the video file before 2002/12/24 23:59:59-->
    <grant>
      <mx:record/>
      <mx:diReference>
        <mx:identifier>urn:uuid:14324da6-7d37-49c8-9656-05acadfd2b9c</mx:identifier>
      </mx:diReference>
    </grant>
    <grant>
      <mx:play/>
      <mx:diReference>
        <mx:identifier>urn:uuid:14324da6-7d37-49c8-9656-05acadfd2b9c</mx:identifier>
      </mx:diReference>
      <validityInterval>
        <notAfter>2002-12-24T23:59:59</notAfter>
      </validityInterval>
    </grant>
    <issuer>
      <!-- The issuer's signature -->
      <dsig:Signature>
        <dsig:SignedInfo>
          <dsig:CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-
20010315"/>
          <dsig:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
          <dsig:Reference>
            <dsig:Transforms>
              <dsig:Transform Algorithm="urn:mpeg:mpeg21:2003:01-REL-R-
NS#license"/>
            </dsig:Transforms>
            <dsig:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
            <dsig:DigestValue>PB4QbKQOC0941tTEbj1/Q==</dsig:DigestValue>
          </dsig:Reference>
        </dsig:SignedInfo>
        <dsig:Signature Value>allDoeplvDzDWqZU+7Tdt4kg7DRt9bu5K9I6p5C32wsNb6kUNJ4q9sH/
2OLPTHsUnTuPdaGncWhJRyoYRjJqA==</dsig:Signature Value>
        <dsig:KeyInfo>
          <dsig:KeyValue>
            <dsig:RSAKeyValue>
              <dsig:Modulus>g8NRYMG307NqJgmZG8TIUOp+9sQjsAai+hLpkBiLaf4RvhvS3pD0dvy1YosEjKL8mk/
KTGniC+pY4ia5kLByQ==</dsig:Modulus>
              <dsig:Exponent>AQABAA==</dsig:Exponent>
            </dsig:KeyValue>
          </dsig:KeyInfo>
        </dsig:Signature>
      <!-- Additional details about the license issuance, such as the issued time -->
      <details>
        <timeOfIssue>2002-01-27T15:30:00</timeOfIssue>
      </details>
    </issuer>
  </license>
</licenseGroup>

```

The following demo shows two difference cases. The first case is that the user wants to play the video file at 2001/12/23 01:30:00. The user's query file is shown below.

```

<?xml version="1.0"?>
<query xmlns="http://www.contentguard.com/2002/exampleQueryNS">
  <goalToCheck>
    <principalToCheck
      rsaModulus="Fa7wo6NYfmvGqy4ACSWcNmuQfbcjSZx7aCibIgkYswUeTCrmS0h27GJrA15SS7TYZzSfaS0xR9IzdUEF0ThO4w=="
      rsaExponent="AQABAA==" />
    <rightToCheck>
      <mx:play xmlns:mx="urn:mpeg:mpeg21:2003:01-REL-MX-NS" />
    </rightToCheck>
    <resourceToCheck uri="urn:uuid:14324da6-7d37-49c8-9656-05acadfd2b9c" />
  </goalToCheck>
  <simulationExerciseEnvironment>
    <time>2001-12-23T01:30:00</time>
  </simulationExerciseEnvironment>
</query>

```

Figure 5-16 shows that the authorization result is successful after examining the license xml file and the query file. So, the video file can be rendered. Figure 5-17 is the screenshot during playing back. Figure 5-18 is the dialog which shows the information about the authorization.

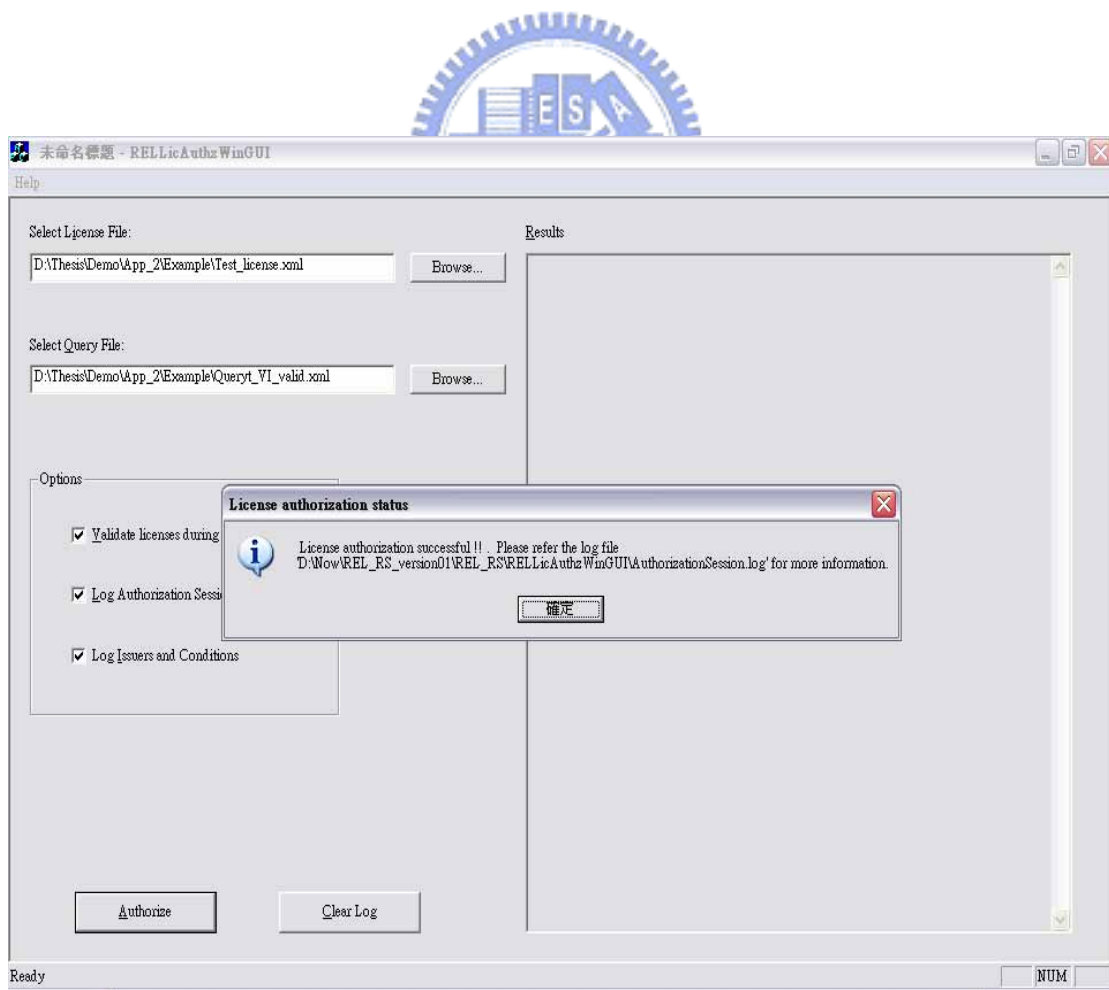


Figure 5-16 Authorization result dialog



Figure 5-17 The screenshot

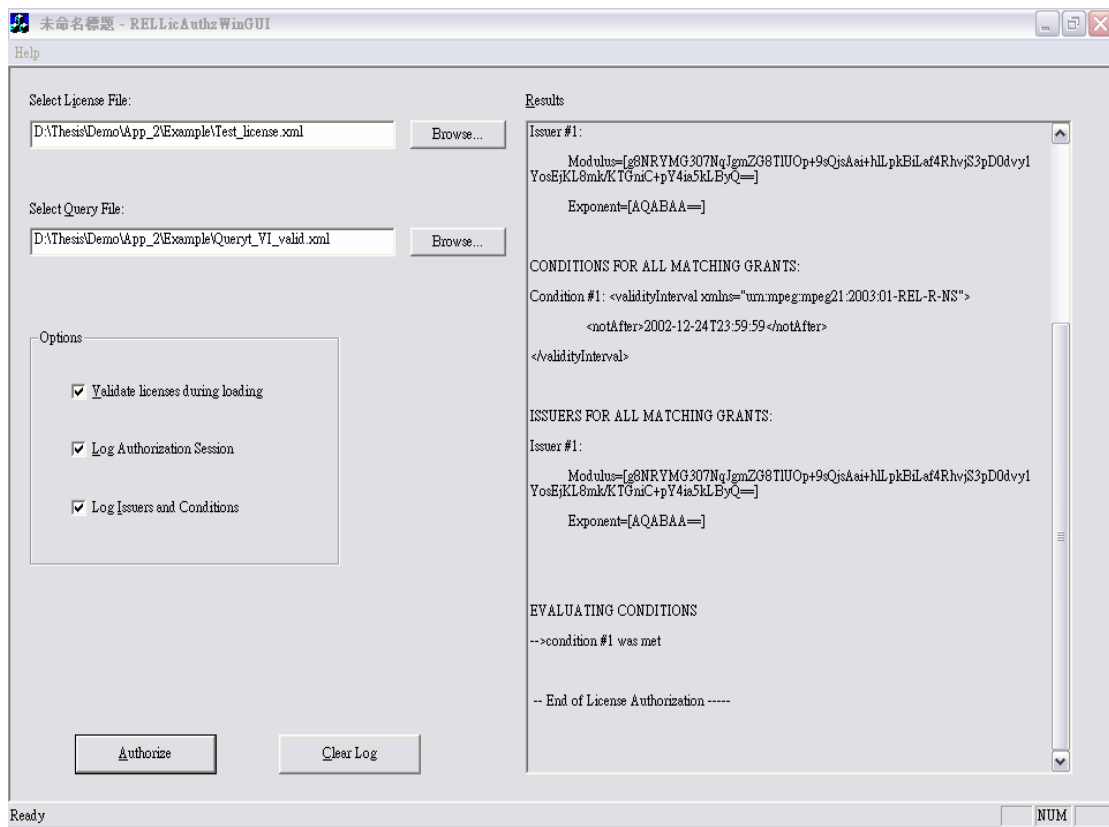


Figure 5-18 Log of authorization

The second case is that the user wants to play the video file at 2003/12/23 01:30:00. The query file is shown below.


```

<?xml version="1.0"?>
<query xmlns="http://www.contentguard.com/2002/exampleQueryNS">
  <goalToCheck>
    <principalToCheck
rsaModulus="Fa7wo6NYfmvGqy4ACSWcNmuQfbcjSZx7aCibIgkYswUeTCrmS0h27GJrA15SS7TYZzSfaS0xR9IzdUEF0ThO4w=="
rsaExponent="AQABAA=="/>
    <rightToCheck>
      <mx:play xmlns:mx="urn:mpeg:mpeg21:2003:01-REL-MX-NS"/>
    </rightToCheck>
    <resourceToCheck uri="urn:uuid:14324da6-7d37-49c8-9656-05acadfd2b9c"/>
  </goalToCheck>
</simulationExerciseEnvironment>
  <time>2003-12-23T01:30:00</time>
</simulationExerciseEnvironment>
</query>

```

Figure 5-19 shows that authorization result is unsuccessful. So, the video file can not be rendered. Figure 5-20 is the dialog which shows the information about the authorization.

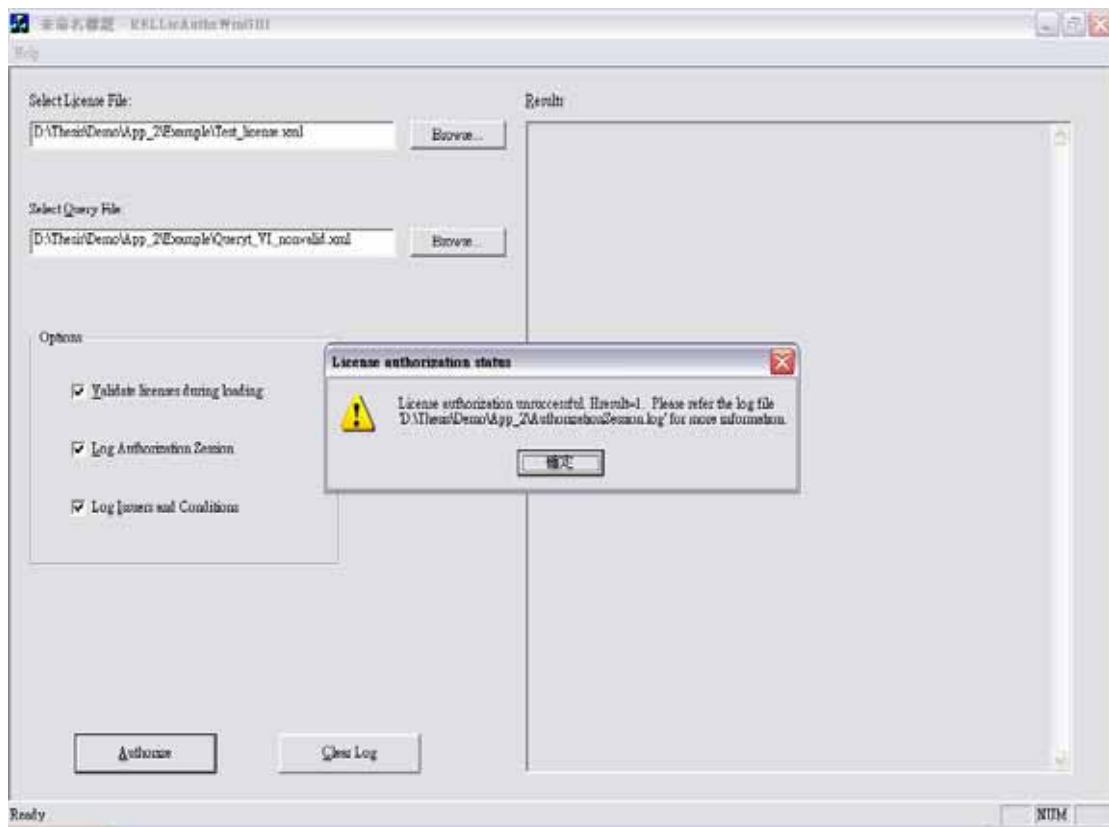


Figure 5-19 Authorization result dialog

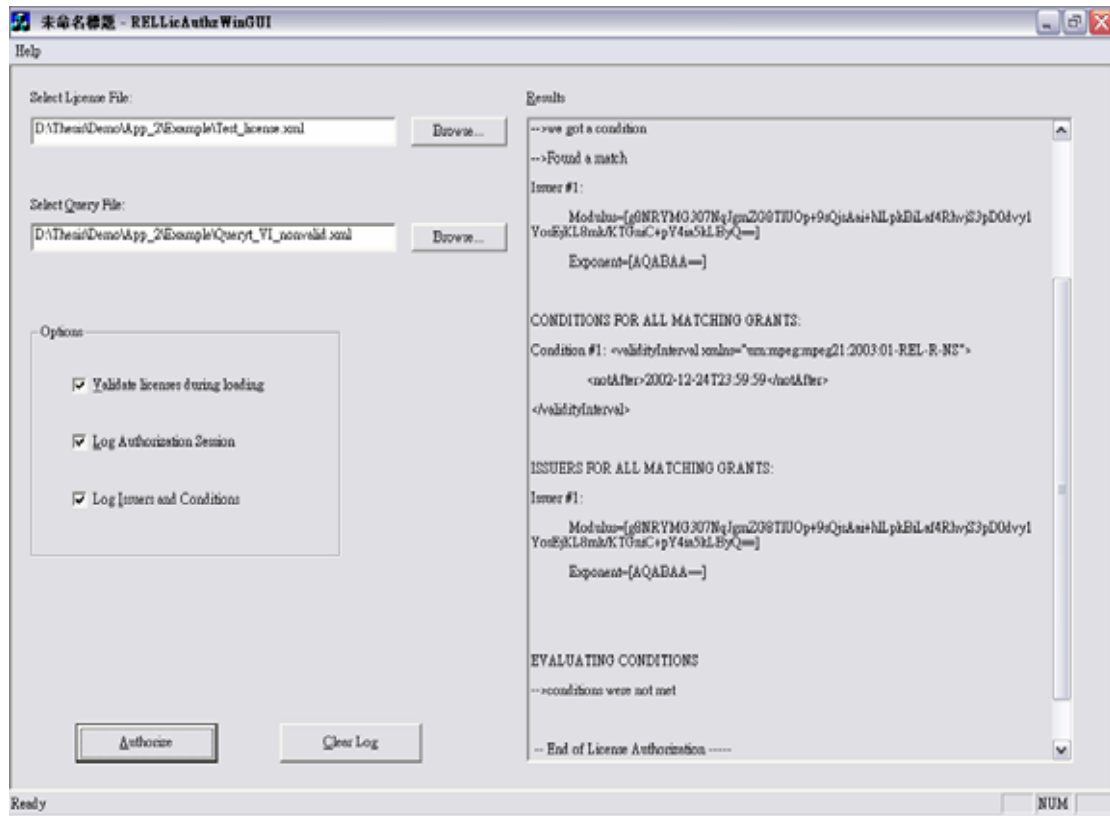


Figure 5-20 Log of authorization



Chapter 6

Conclusion

6.1 Contributions

This goal of this thesis is to study MPEG-21 Rights Expression Language (REL) and Service Information (SI) in DVB system and how we use MPEG-21 REL to construct practical Digital Rights Management (DRM) system from DVB platform. The use of the Twinhan DVB-T PCI Card makes our experimental system close to practical systems. Our emphasis here is our design and implementation of an MPEG-21 REL-based DRM system.

We first design and implement a Conditional Access scheme by using Licenses and the Key management mechanism we proposed in section 5.1.2. In a broadcasting environment, the license is transmitted along with the DVB content. The content provider or the service provider can also use the license to control the usage of contents on different DRM systems as long as the same REL syntax is adopted. In the second application, we show how the rights of content can be controlled and enforced even after the contents are downloaded. A trusted device will check the License before playing back a digital item. This License is examined by REL interpreter and authorization module. Therefore, the subscriber behavior on any trusted device is well monitored.

6.2 Future Works

In the current implementation, we design two simple DRM schemes for demonstration. Their functions are rather limited. We could use MPEG-21 IPMP

system as our DRM system. IPMP (Intellectual Property Management and Protection Extension) system is a Digital Rights Management interface and architecture specifications developed by the MPEG group. The IPMP system allows the interoperability of the DRM system. By integrating the MPEG-21 REL parser into the MPEG-4 IPMPX system, we could increase the security robustness and interoperability.



Reference

1. *Information Technology - Generic Coding of Moving Pictures and Associated Audio: Systems Recommendation H.222.0*, ISO/IEC JTC1/SC29/WG11, N0801, November 1994.
2. *Digital Video Broadcasting (DVB) ,Specification for Service Information (SI) in DVB systems*, ETSI EN 300 468 V1.4.1, November 2000.
3. Interactive TV Web, <http://www.interactivetvweb.org/index.shtml>
4. *Text of ISO/IEC 21000-5 FCD — Part 5: Rights Expression Language*, ISO/IEC JTC 1/SC 29/WG 11/N5349, December 2002, Japan.
5. ContentGuard, Inc.. (2001) eXtensible Rights Markup Language (XrML) 2.0 Specification, <http://www.xrml.org>
6. X. Wang, T. DeMartini, B. Wragg, M. Paramasivam, and C. Barlas, “The MPEG-21 Rights Expression Language and Rights Data Dictionary,” *IEEE Multimedia*, vol. 7, no. 3, pp. 408-417, June 2005.
7. J. Bormans and K. Hill, “MPEG-21 Overview v.5,” ISO/IEC JTC 1/SC 29/WG11 N5231, Shanghai, October 2002.
8. “Introducing MPEG REL - an Overview,” ISO/IEC JTC 1/SC 29/WG11 N7427, Poland, July 2005.
9. W3C. (1999) Namespaces in XML, <http://www.w3.org/TR/1999/REC-xml-names-19990114>
10. W3C. (2001) XML Schema, <http://www.w3.org/TR/2001/REC-xmlschema-1-20010502>
11. *Information technology — Learning, education, and training — Profiles of standards and specifications — Part xxx: Profile of Rights Expression Language (REL)*, ISO/IEC JTC 1/SC 36/WG 4/N0113, US, August 2004.
12. *Information Technology – Multimedia Framework – Part 6: Rights Data Dictionary*, ISO/IEC 21000-6:2004, 2004.
13. *Information Technology – Multimedia Framework – Part 3: Digital Item Identification*, ISO/IEC 21000-3:2003, 2003.
14. *Information Technology – Multimedia Framework – Part 2: Digital Item Declaration*, ISO/IEC 21000-2:2003, 2003.
15. An Overview of Cryptography, <http://www.garykessler.net/library/crypto.html>
16. W. Diffie and M.E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644--654, November 1976.
17. R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public key cryptosystems. *Communications of the ACM*, 21(2):120--126, February 1978.
18. DirectX 9.0 Documentation for C++, Microsoft Corporation, 2002.
19. Kivco Consulting Inc., <http://www.kivco.com/securitydoc.html>
20. Xin Wang, et al., “An Example Implementation of MPEG-21 REL Reference

- Software,” ISO/IEC JTC1/SC29/WG11 MPEG2003/M9581, March 2003
21. Y.-L. Peng, “An Implementation of Multimedia Content Guard Based on MPEG-4 IPMP-X System,” M.S. thesis, Dept. Computer Science, National Chiao Tung University, Hsinchu, Taiwan, R.O.C., June 2004.
 22. EEdesign, <http://www.eedesign.com.tw/article/forum/fo700.htm>.

