國立交通大學

資訊科學與工程研究所

碩士論文

高等數位內容存取系統
運用架設於公鑰基礎設施的物件認證服務

Advanced Access Content System using
PKI based Authentication Service

研究生：李英宗

指導教授：葉義雄 教授

中華民國 九十五 年七月

中文摘要

隨著數位時代的來臨，以及網際網路的普及，非授權影音內容的散播變得比以往更加容易。也因此，個人或是企業所創造之智慧財產，也較以往更容易受到侵犯。為因應內容數位化之潮流與趨勢，娛樂產業對於數位智財管理技術之需求更勝以往，以期能更進ㄧ步保護數位影音作品之價值及創作者之權益。

光學儲存媒介，如 CD，DVD 等，為目前市面上最受歡迎之影音內容儲存方式。科技的日新月異，更已造就在儲存空間、影音品質等方面都大幅提升之新世代藍光儲存技術。本篇論文所探討之「Advanced Access content System」，或簡稱「AACS」，可翻成「高等數位內容存取系統」，為一逐漸成型之新世代防拷技術，意在提供影視產業在散佈光學儲存產品之同時，亦可防止非授權盜版之機制。雖然在設計上 AACS 已包含數項標準化之加密程序，多項創新專有之設計亦相形降低 AACS 在與其他系統整合上之彈性。另外，在公開密鑰憑證 (Public Key Certificates) 方面，AACS 並無提供標準化之規格，這可能會增加此技術在推廣上之困難度。在介紹 AACS 之同時，本篇論文意圖闡述將 X.509 架構納入AACS規格之優點，並提供AACS在規格修正上之建議。

關鍵字: 高等數位能容存取系統、X.509 公鑰憑證架構、公鑰基礎設施。

Abstract

Digital technologies have given entertainment industries tremendous opportunities for content creation. However, they have also granted anyone the ability to create exact duplications and to distribute these copies with ease that facilitates copyright infringement that reduces the revenue inflow for those creative minds. Hence, there is a great desire for digital rights management (DRM) systems that can preserve the economic value of digital video and protect the rights of its owners.

Optical storage media have become the most popular vehicle of carrying audiovisual content after their theatrical debut. The advancement of technology brought us with each new generation increased storage capacity and improved picture and sound quality. Advanced Access Content System is a copy protection mechanism proposed to protect the audiovisual entertainment content stored on the next generation optical storage media. Although the specification has used several standardized cryptographic procedures in its design, several proprietary design decisions may make AACS less flexible and more difficult to integrate with other systems. In particular, the public key certificates in AACS are not defined using any standardized format. This thesis will attempt to suggest some modifications to the AACS specification that incorporate X.509 certificate framework. Some benefits of doing so are also discussed.

Keywords: AACS, X.509, PKI.

致　謝

能夠順利完成這篇論文，首先要感謝我的指導老師葉義雄教授，在這兩年內，給予我莫大的空間與自由，讓我在完成碩士學業之餘，也能順利通過博士資格考，使我在下個階段的學業生涯上，沒有後顧之憂。也非常感謝口試委員—蔡文能教授及周勝鄰教授—對於我的論文的不吝指教，可以讓我把論文修改的較為完善。尤其得感謝蔡文能教授的熱心支持，沒有您緊迫釘人的督促，我想這篇論文至今仍無法成形。在口試前晚陪我一起熬夜修正口試文稿的恩情，我至今仍銘記于心。

此外，還得感謝建興電子的賴遠青先生，在報告技巧的方面，給予我許多修正及改善的意見，讓我受益良多。實驗室的眾多成員，如陳以德、黃定宇、李鎮宇、高銘智學長，甘老大、白台柱、昇哥、許鴻祥同窗，Gobby、伯昕以及胖婷學弟妹都在我碩士生涯中給予我默默的支持與鼓勵。

最後，我必須要感謝我的家人多年來給我的支持與栽培，以及關懷與勉勵，讓我在學習之路能夠堅持至今。謝謝你們。

詞窮不及達意，族繁不及備載。在這麼短的篇幅中，如有遺漏，在所難免。在這邊，僅將此論文獻給所有關心、支持我的人。

李英宗

中華民國九十四年七月

Table of Content

List of Tables

List of Figures

Chapter 1 Introduction

Digital technologies have given entertainment industries tremendous opportunities for content creation. However, they have also granted anyone the ability to create exact duplications and to distribute these copies with ease that facilitates copyright infringement often referred to as “piracy”. KaZaA [12], BitTorrent [13], eDonkey [14], and Gnutella [15] are popular Peer-to-Peer (P2P) software that has been used to share copyrighted music, movies, software and other materials. Future P2P systems may encrypt shared data, preserve user anonymity, support a larger user base, and improve its robustness [21]. Although these systems may have legal usages, content creators and owners are mostly concerned about the consequences of unauthorized copying and distribution on a massive scale. Hence, there is a great desire for digital rights management (DRM) systems that can preserve the economic value of digital video and protect the rights of its owners.

The Internet is not the only place where digitized audiovisual entertainment products reside. Optical storage media such as compact discs invented in 1980s, although originally developed for storing digital audio only, has long ago become the mainstream storage method for digital video and has revolutionized the entire distribution channel for music and video contents. The inherent “off-line” nature of optical disc distribution method has created unique challenges for copy protection designers and developers. Some previous approaches have not always succeeded. The content scrambling system [16] (CSS) for protecting pre-recorded movies stored on digital video discs (DVD) is one such failed approach most cited in literatures [19][22]. The system intends to keep the video data stored on the disc in an encrypted form, and only entrusts certain secrets to the compliant players that enable them to playback the encrypted video stream. However, not every player can be trusted to maintain the secrecy of the secrets. In 1999, a group of Norwegian programmers successfully reverse-engineered a DVD player software to build a decryption engine aptly named DeCSS. The program quickly spread through the Internet to permit anyone to bypass CSS protection.

Advanced Access Content System (AACS) is one system in development trying to address weaknesses in CSS, and aims to protect the next generation optical storage media. The system incorporates several cryptographic methods to provide security services that could make access control possible. However, some of the AACS cryptographic functions are proprietary. This thesis tries to incorporate a well-established PKI standard to improve AACS's flexibility.

1.1 Optical Media Copy Protection

The Compact Disc (CD), introduced in 1982, was originally developed as an "audio-only" storage medium. The technology was later evolved to hold other information such as video recordings and computer software, and was known as a CD-ROM. Even though the discs are read-only initially, the technology quickly expanded to include the capability to write-once or even to write multiple times (CD-R and CD-RW). At this point, PC users were given an alternative means to store huge amount of data. However, the original specification (The Red Book) for audio disc does not include any serious copy protection mechanism.

In September of 1996, version 1.5 of DVD specification was finalized. Although resembling a compact disc in physical appearance, a DVD has a much higher storage capacity, allowing it to record movies with higher video and audio quality. Depending on its format and structured content, the disc is appropriately referred to as DVD-Video, DVD-Audio, and DVD-Data. DVD-Video has four complementary systems designed to restrict the DVD user in various ways: region codes, Content Scrambling System (CSS), Macrovision, and disabled user operations (UOPs).

One or more region codes are contained in each DVD-Video disc. They are meant to denote the area(s) of the world in which distribution and playback are intended. Commercial DVD players are mandated by their specification to only play discs with a region code that matches its own. The goal was to enable region-by-region control of the various aspects of a release such as content, date and price by the motion picture studios. However, many DVD players, or their modifications, are able to playback video discs of any region. Since almost all televisions in Europe, Australasia, and Taiwan are capable of displaying NTSC video, the consumers in these regions have a huge choice of discs. Region codes thus lose its intended purpose on the worldwide scale.

Content Scrambling System is a complex system with many components to impede illegal copying of CSS-protected video discs. Its components include encryption to scramble video data written on the discs, a protocol for obfuscating the communications between the DVD reader and attached devices, and copy protection for digital and analog outputs. CSS-compliant readers are given some secret information that enables them to access the decryption keys stored on the disc. However, the successful discovery of the CSS encryption algorithm by a group of Norwegian programmers through the art of hacking and reverse-engineering led to the development of a software program that can decrypt any CSS-protected DVD. The program is appropriately named DeCSS [22]. It processes a CSS-encrypted DVD disc and produces an unencrypted movie that can be copied and distributed without any restriction. Any DVD reader, including readers that do not recognize CSS protection, can be used to playback these copies.

Macrovision is a video copy prevention scheme created by a company with the same name. The technology makes use of the off-screen region of a video signal to implant special signals that could prevent DVD recorders from correctly recording encoded video. The recorded video will appear scrambled or fade between overly light and dark.

DVD-Video also provides a way for the content owner to specify the set of operations that a user is allowed to perform. For example, the disc may prohibit the user from selecting a menu, skipping chapters, forwarding or rewinding. This feature is known as User Operation Prohibitions, or Prohibited User Operations (UOPs or PUOs). However, this feature only works when the DVD player respect these commands. Like region codes, DVD players can be modified to ignore the usage rules demanded by UOPs.

With the development of next generation optical discs, an effective copy protection mechanism is highly sought after. The Advanced Access Content System (AACS) is a standard in development that has been adapted to Blu-ray Disc and HD DVD. The standard is purported to provide content owners the ability to restrict access to and copying of their multimedia content.

The proposal is based on the broadcast encryption model originated from digital TV broadcasting domain. Analogous to a digital TV broadcast, the broadcasted video content reaches players in the form of optical discs. The idea is to only grant legitimate players the viewing privileges. The governing authority reserves the right to revoke a device if that device is discovered to be compromised.

1.2 Motivation

In the summer of 2004, a group of companies, which include Disney, Intel, Microsoft, Matsushita (Panasonic), Warner Brothers, IBM, Toshiba, and Sony, have joined forces to create and administer a standard called Advanced Access Content System (AACS). The standard has been adopted as the copy protection mechanism for HD DVD and Blu-ray Disc. The two next generation optical storage technologies incorporates a shorter wave-length laser light that could record onto and read more data out of a plastic with the same size as a DVD disc.

AACS combines several cryptographic concepts to control the access to the media. The technology borrows the idea of broadcast encryption to allow only legitimate devices to retrieve the content through subset difference trees. The content encrypting keys would be embedded in the disc itself through a cryptographic process. Legitimate devices would be given enough secret information to obtain the keys that would allow them to view the content. During the decrypting process, AACS would make use of self-defined certificates to either verify the authenticity of the participating devices and to check the integrity of several control information.

However, the proprietary certificate formats might make AACS less flexible and less interoperable with other components of DRM systems. In this thesis, we study the possibility of using a standard X.509 certificate framework to restructure AACS in order to lift these limitations.

1.3 Thesis Organization

Chapter 1gives a brief overview of the optical media industry, and describes some problems occurred in previous copy protection mechanism. This chapter also provide research motivations the Advanced Access Content System for the first time. Before we explain the Advanced Access Content System in detail, we first provide the background knowledge required to understand the concepts used in the specification in Chapter 2. Then, Chapter 3attempts to consolidate the AACS v0.91 specification, which is separated into 7 “books”, into a single chapter. Due to the shear amount of information, the focus will be on the format independent part of the specification. In Chapter 4, we propose a modification to the AACS design that could enhance the system’s flexibility. Finally, we conclude the thesis and make a few suggestions on the future direction to pursue in Chapter 5.

Chapter 2 Background Knowledge

Advanced Access Content System, the subject matter being studied in this thesis, incorporates many concepts in the fields of cryptography and information security in general to provide its purported copy-protection functions. The system can also be fit into a larger Digital Rights Management (DRM) framework. This chapter attempts to give an abridged background knowledge involved in the development of AACS. Also, the related technologies employed in our proposed scheme are also introduced in this chapter.

2.1 Optical Storage Media Format

For thousands of years, spoken words can only be passed down the generations through the scratches of shells, the carvings of bamboo sheets, and the smudges of dried pulp. Other sounds, natural or artificial, can only be remembered through the heart of the listener. The invention of phonograph by Thomas Edison in 1977 changed all that. People do not have to be told of a magnificence of a masterpiece; the masterpiece can be heard, again and again. Although the technique of Edison had been improved from recording the physical vibration directly onto a wax plate to magnetically onto a flexible strip, the analog nature of the process makes the recording prone to the effect of ever-present noise. The historical route taken by the humankind to discover a way to make permanent our visual perception is nothing but a bumpy road. It took several decades to bring color television, VCRs, and video camcorders into the hands of an average citizen.

At the inception of the digital era, the need to store huge amount of binary data created by a video stream naturally arises. The portability and the storage capacity of an optical media nicely provide a solution. In this section, we will look at some major optical media formats that have evolved over the years.

2.1.1 Compact Disc

A Compact Disc (or CD), introduced in 1982, is an optical disc originally developed to store digital audio. Commercial record labels continue to produce most of their audio recordings on CDs as of mid-2006. Standard compact discs come in two sizes. Compact discs measuring 12-cm in diameter are probably the most familiar varieties to us, and they can hold approximately 80 minutes of audio. The smaller 8-cm discs can hold approximately 20 minutes of audio only, and thus are sometimes used for CD singles.

Compact disc technology was later adapted to produce discs that can hold other forms of data. In addition, consumers are given the ability to store their own data on compact discs with the addition of record-once and rewritable media (CD-R and CD-RW) into the format family. The CD and its extensions have been extremely successful not only in the consumer electronics industry but also in the personal computer arena even up to this day.

History

In 1979, Philips and Sony decided to join forces, setting up a joint task force of engineers whose mission was to design the new digital audio disc. After a year of experimentation and discussion, the task force produced the “Red Book”, the Compact Disc standard. The Compact Disc reached the market in late 1982 in Asia and early the following year in other markets. This event is often seen as the “Big Bang” of the digital audio revolution, and the new audio disc was enthusiastically received. From its origins as a music format, Compact Disc has grown to encompass other applications. In June 1985, the CD-ROM (read-only memory) and, in 1990, CD-Recordable were introduced, also Developed by Sony and Philips.

Physical Details

Compact discs are made from a 1.2 mm thick pure polycarbonate plastic disc with a thin layer of super purity Aluminum applied, and then protected by a film of lacquer, which can be printed with a label. Binary data is stored onto a compact disc as a series of microscopic indentations situated in a tightly packed track molded into the top of the polycarbonate layer. A CD is read by focusing a 780 nm wavelength semiconductor laser through the bottom of the polycarbonate layer. The difference in height between I indented and un-indented areas leads to a phase difference in the light reflected. By measuring the reflected intensity with a photodiode, the disc drive is able to tell the hills from the valleys. However, the pits and lands themselves do not represent the zeroes and ones of binary data. Instead, a change in the heights indicates a one, while no change indicates a zero. Finally, the modulation and coding process used in mastering the disc is reversed to reveal the raw audio data stored on the disc.

Standard CDs are available in two sizes. By far, the most common is 12 cm in diameter, with either a (74 min./650 MB) or a (80 min./700 MB) capacity. Eight centimeter discs are also available. They are mainly used for audio CD singles in some regions. Those smaller CDs can hold 21 minutes of music, or 184 MB of data. Other non-standard shapes and smaller form factors have also been sold or given away as promotional items.

Recordable CDs

Recordable compact discs, CD-Rs, are molded with a "blank" data spiral applied with a photosensitive dye. Then, the discs are metallized and lacquer coated. The write laser of the CD recorder changes the color of the dye to allow the read laser to “see” the data. CD-R recordings are permanent. The resulting discs can be read by most CD-ROM drives and played in most audio CD players.

CD-RW is a re-recordable medium that uses a metallic alloy instead of a dye. The write laser in this case is used to heat and alter the chemical properties of the alloy and hence change its reflectivity. A CD-RW does not have as great a difference in the reflectivity of lands and bumps as a pressed CD or a CD-R, and so many CD audio players cannot read CD-RW discs, although the majority of stand-alone DVD players can.

Copy Protection

The Red Book audio specification does not include any serious copy protection mechanism, except to have a simple ‘anti-copy’ bit in the subcode. Starting in early 2002, attempts were made by record companies to market “copy-protected” non-standard compact discs. However, the public at large are greatly against these copy-protected discs because many see them as a threat to fair use.

2.1.2 Digital Versatile Disc

In early 1990s, Phillips and Sony created a joint venture to develop a new optical storage technology. The result was the DVD specification version 1.5, announced in 1995 and finalized in September 1996. DVDs resemble compact discs with the same physical dimensions, but they have different encoding format and can store much higher quality visuals and sounds. The official DVD specification is now maintained by the DVD Forum [11].

History

In the early 1990s, two high density optical storage standards were being developed. MultiMedia Compact Disc (MMCD) were backed by Philips and Sony while the other standard, Super Density disc (SD), were supported by Toshiba, Time-Warner, Matsushita Electric, Hitachi, Mitsubishi Electric, Pioneer, Thomson, and JVC.

Philips and Sony abandoned their MMCD format and agreed upon Toshiba's SD format with two modifications. The end result was the DVD specification version 1.5, announced in 1995 and finalized in September 1996. In May 1997, the DVD Consortium, founded by ten companies including Philips and Sony, was replaced by the DVD Forum, and now has over 200 member companies.

“DVD” originally stands for “Digital Video Disc”. Some members of the DVD Forum believe that it should stand for "Digital Versatile Disc" to reflect its widespread use for non-video applications. However, the DVD Forum never reached a consensus on the matter, and so today the official name of the format is simply “DVD” [20].

The first DVD players and discs were available in 1996 for Japan, in 1997 for the United States, in 1998 for the European countries and in 1999 for Australia. The first film released in DVD format was Twister in 1996.

DVD rentals first surpassed those of VHS during the week of June 15, 2003. In June 2005, several retailers in U.S. announced plans to phase out the VHS format entirely, in favor of the more popular DVD format. Consumers have predicted that 2006 would be the final year for new releases on VHS.

DVD-Video

DVD-Video is the application format designed to store movies on a DVD disc. The video images are encoded in compressed MPEG-2 file format and the soundtrack is encoded in a variety of audio formats. The storage capacity of a DVD-Video disc allows a wide variety of extra features in addition to the feature film itself. The extra features can include director’s commentary, closed caption and dialogues in multiple languages, extra scenes cut from the theatrical distribution, behind the scenes documentary and simple interactive games.

Other extras comprise animated menus, still pictures, branching for multiple storylines, and views from different camera angles. Also, DVD-Video discs can store additional data files that only can be read by computer DVD drives.

Restrictions

Four complementary systems are designed to restrict DVD-Video users in various ways: Macrovision, Content Scramble System (CSS), region codes, and disabled user operations. Macrovision is a video encoding technology developed by Macrovision Corporation which was established in 1983. The technology would cause VCRs unable to record video streams received from DVD discs encoded with Macrovision. The recording would appear to be scrambled, or else the images would flicker upon playback. CSS utilizes a stream cipher to encrypt the video data to discourage illicit copying. However, a public-domain software called DeCSS has been developed that would render CSS useless. Region codes are designed to restrict the distribution of a

DVD title within a geographic region. For example, a Taiwanese movie is not accessible on a U.S. DVD player. Furthermore, content providers can specify certain user operations to disable to prevent users from skipping over copyright notice, for instance. However, the last two restriction mechanisms require the cooperation of DVD players.

2.1.3 HD DVD

On November 19, 2003, the DVD Forum decided to adapt HD DVD, previously called the "Advanced Optical Disc (AOD)", to be the DVD successor for HDTV. HD DVD (High Density DVD or High Definition DVD) is a digital optical media format with the same physical size as CDs and DVDs which can hold high definition video or other data. HD DVD uses 405 nm wavelength blue laser. The current specification version for HD DVD-ROM and HD DVD-Rewritable is version 1.2. The specification for HD DVD-RAM is currently at 2.2. HD DVD is promoted by Toshiba, NEC, Sanyo, Microsoft, and Intel, among others. The format is currently backed by major studios such as Universal Studios, Paramount Pictures, Warner Brothers. Studio Canal, and The Weinstein Company. Japanese were the first to see HD DVD players, which were released by Toshiba on March 31, 2006, and Americans followed shortly after on April 18, 2006.

2.1.4 Blu-ray Disc

A Blu-ray Disc (BD) is currently competing with HD DVD format for wide adoption as the preferred next-generation optical disc format meant for high-density storage of high-definition video and data. The Blu-ray standard was jointly developed by the Blu-ray Disc Association (BDA) which consists of a group of consumer electronics and PC companies, spearheaded by Sony. As of 2006, neither format has succeeded in superseding the present home video standard, the DVD.

The name Blu-ray is derived from the blue-violet read-write laser. The shorter wavelength (405 nm) of the blue-violet laser allows more information to be stored digitally in the same amount of space, which gives a Blu-ray Disc much higher storage capacity than a DVD disc. In comparison to HD DVD, Blu-ray has more information capacity per layer but may initially be more expensive to produce.

The Blu-ray Disc Association unveiled their plans for a May 23, 2006 release date at the Consumer Electronics Show (CES) in January 2006. Since then, Blu-ray was delayed, but finally shipped in the U.S. on June 20, 2006.

2.2 Copy Protection Concepts and Issues

Digital audio, video, and images have gained importance as a source of information for professional use and entertainment. Current digital multimedia content has a predefined file format and can be stored and distributed on every medium that can carry digital information. New content creation strategies are developed as content quality, network bandwidth, and digital storage capacity have continued to improve. At the same time they create exciting new products for consumers, content industry is faced with ever increasing threats of lost revenues due to unauthorized copying and distribution of content. Modern computer technologies make it extremely efficient and cost effective to duplicate and distribute multimedia content over the Internet. As a result of this, content owners and distributors are desperately in need to defend their property right and revenue by deploying content protection technologies in an attempt to prevent the unauthorised use of multimedia content.

Numerous content protection technologies and applications have been devised based on watermarking and cryptographic principles [27]. Most protection methodologies focus on technical aspects that are rigid and inflexible, which result in the expense of user friendliness and privacy. Moreover, protected contents may need to be converted to an unprotected format before they can be transferred to an intrinsically untrustworthy device for viewing. Hence, 100% copy protection is believed to be an unattainable end [28].

Nevertheless, this section provides a glimpse of techniques or concepts that have been incorporated into the design of copy protection schemes. Also, this section will give a list of issues in providing digital copy protection.

2.2.1 Broadcast Encryption

Broadcast encryption is the cryptographic problem of encrypting broadcast content in such a way that only qualified users can decrypt the content. The challenge arises from the dynamically changing set of qualified users. Users can become qualified by subscribing and they can become illegitimate by not paying the fee. Any solution to such an event should not affect the users already qualified or unqualified. Several solutions exist offering various tradeoffs between the overhead in the broadcast, the number of keys that each user needs to store, and the feasibility of decryption by a collusion of unqualified users.

However, no solution can prevent rogue users from sharing their decryption keys with unqualified users. These rogue users are called traitors in the literature. Therefore, Traitor tracing algorithms are required to retroactively identify the traitors so countermeasures can be taken to minimize the damage. In practice, Pay TV systems often employ set-top boxes with tamper-resistant smart cards that prevent a user from learning their own decryption keys.

Multimedia contents distributed on optical discs share many problems as television programs distributed over digital broadcast. Content owners would like to grant compliant playback devices the privilege to access the contents stored on the discs while deny other devices from the access. They would also like to revoke users from viewing privilege once they discovered the users had violated certain usage rules. Hence, solutions to the broadcast encryption problem may also be adapted to protect content owners from unauthorized copying of their properties.

AACS, the copy protection scheme studied in this thesis, uses a broadcast encryption algorithm that enables the provider to revoke any desired subset of users from the content access by rearranging keys in a binary tree. Each user device has its place in the leaves of the binary tree. The keys are assigned to the device in such a way that the device could not learn the keys on the node from its corresponding leave nodes to the root node. If the master secret is encrypted with one of those keys, the device will not be able to decrypt it. Hence, the device is effectively prevented from viewing the content.

2.2.2 Traitor Tracing

Traitor tracing is a copy prevention strategy which has been around for years. The main concept is that each user device is given a unique key or a unique set of keys. The key(s) are combined with the content encryption key(s) in a certain way to write some signature data on the copy made with that device. If content decryption key is made public, content owners can examine the illegal copy to determine the traitor device from their database of assigned codes.

2.3 Cryptography Primer

2.3.1 Symmetric Cryptography

Symmetric ciphers are cryptographic algorithms that encrypt and decrypt messages using the same secret key. These algorithms require that the sender and receiver agree on a key before they can communicate securely. The security of a symmetric algorithm rests in the secrecy of the key; compromising the key means that anyone could encrypt and decrypt messages.

Symmetric ciphers can also be divided into two categories: symmetric block cipher and symmetric stream cipher. Symmetric block cipher encrypts a plaintext block into a ciphertext block with an equal length. Symmetric stream cipher, on the other hand, sees messages as a bit stream or a byte stream and encrypts them accordingly.

2.3.2 Mode of Operation

Symmetric block cipher algorithms are designed to operate on fixed length message blocks. National Institute of Standards and Technologies (NIST) has published a special publication 800-38A [10] detailing five ways a block cipher can be used to encrypt/decrypt longer messages.

- Electronic Code Book Mode

Electronic code book (ECB) mode is the most obvious way to use a block cipher. A message is divided into blocks of equal length and each block of plaintext encrypts into a block of ciphertext.

ECB Encryption: $C_j = CIPH_K(P_j)$ for $j = 1, \ldots, n$

ECB Decryption: $P_j = CIPH_K^{-1}(C_j)$ for $j = 1, \ldots, n$

Although this mode is simple, it is not safe to use it to process significant amount of information since identical plaintext blocks would encrypt to identical ciphertext blocks.

- Cipher Block Chaining Mode

In cipher block chaining (CBC) mode, the plaintext is XORed with the previous ciphertext block before it is encrypted. The first plaintext block is XORed with an extra initialization vector (IV) block before it is encrypted.

CBC Encryption: $C_1 = CIPH_K(IV \oplus P_1)$

$C_j = CIPH_K(C_{j-1} \oplus P_j) \qquad \text{for } j = 2, \ldots, n$

CBC Decryption: $P_1 = IV \oplus CIPH_K^{-1}(C_1)$

$P_j = C_{j-1} \oplus CIPH_K^{-1}(C_j) \qquad \text{for } j = 2, \ldots, n$

- Cipher Feedback Mode

The Cipher Feedback (CFB) mode is a confidentiality mode that features the feedback of successive ciphertext segments into the input blocks of the forward cipher to generate output blocks that are exclusive-ORed with the plaintext to produce the ciphertext, and vice versa. The CFB mode also requires an IV as the initial input block. The CFB mode also requires an integer parameter, denoted *s*, such that $1 \le s \le b$. In the specification of the CFB mode below, each plaintext segment ($P_j^\#$) and ciphertext segment ($C_j^\#$) consists of *s* bits.

CFB Encryption: $I_1 = IV$

$I_j = LSB_{b-s}(I_{j-1}) \,||\, C_{j-1}^\# \qquad \text{for } j = 2, \ldots, n$

$O_j = CIPH_K(I_j) \qquad \text{for } j = 1, \ldots, n$

$C_j^\# = P_j^\# \oplus MSB_s(O_j) \qquad \text{for } j = 1, \ldots, n$

CFB Decryption: $I_1 = IV$

$$I_j = LSB_{b-s}(I_{j-1}) \| C^{\#}_{j-1} \qquad \text{for } j = 2, \ldots, n$$

$$O_j = CIPH_K(I_j) \qquad \text{for } j = 1, \ldots, n$$

$$P^{\#}_j = C^{\#}_j \oplus MSB_s(O_j) \qquad \text{for } j = 1, \ldots, n$$

- Output Feeback Mode

The Output Feedback (OFB) mode is a confidentiality mode that features the iteration of the forward cipher on an IV to generate a sequence of output blocks that are exclusive-ORed with the plaintext to produce the ciphertext, and vice versa.

OFB Encryption: $I_1 = IV$

$$I_j = O_{j-1} \qquad \text{for } j = 2, \ldots, n$$

$$O_j = CIPH_K(I_j) \qquad \text{for } j = 1, \ldots, n$$

$$C_j = P_j \oplus O_j \qquad \text{for } j = 1, \ldots, n-1$$

$$C^*_n = P^*_n \oplus MSB_u(O_n)$$

OFB Decryption: $I_1 = IV$

$$I_j = O_{j-1} \qquad \text{for } j = 2, \ldots, n$$

$$O_j = CIPH_K(I_j) \qquad \text{for } j = 1, \ldots, n$$

$$P_j = C_j \oplus O_j \qquad \text{for } j = 1, \ldots, n-1$$

$$P^*_n = C^*_n \oplus MSB_u(O_n)$$

- Counter Mode

The Counter (CTR) mode is a confidentiality mode that features the application of the forward cipher to a set of input blocks, called counters, to produce a sequence of output blocks that are exclusive-ORed with the plaintext to produce the ciphertext, and vice versa. The sequence of counters must have the property that each block in the sequence is different from every other block. This condition is not restricted to a single message: across all of the messages that are encrypted under the given key, all of the counters must be distinct.

Given a sequence of counters $T_1, T_2, \ldots, T_n$, the CTR mode is defined as follows:

CTR Encryption: $O_j = CIPH_K(T_j)$ for $j = 1, \ldots, n$

$C_j = P_j \oplus O_j$ for $j = 1, \ldots, n-1$

$C_n^* = P_n^* \oplus MSB_u(O_n)$

CTR Decryption: $O_j = CIPH_K(T_j)$ for $j = 1, \ldots, n$

$P_j = C_j \oplus O_j$ for $j = 1, \ldots, n-1$

$P_n^* = C_n^* \oplus MSB_u(O_n)$

2.3.3 Hash Functions

A hash function, *H*(*M*) is a function that operates on an arbitrary length message and returns a fixed-length hash value, *h*.

$H : \Sigma^* \rightarrow \Sigma^n$, *h* = *H*(*M*) where *h* is of length *n*.

Many functions fit the definition of a hash function; however, hash functions need additional “one-way” characteristics to make them suitable to be used in security protocols [23]:

Given *M*, it is easy to compute *h*.

Given *h*, it is computationally infeasible to find an *M* such that *H*(*M*)= *h*.

Given *M*, it is computationally infeasible to find another message, *M’*, such that *H*(*M*) = *H*(*M’*).

Given *h*, it is computationally infeasible to find a pair of messages, *M* and *M’*, such that *H*(*M*) = *h* = *H*(*M’*).

2.3.4 Message Authentication Code

Similar to a one-way hash function, a **message authentication code** (MAC) is a function that processes a variable length message to produce a fixed length message digest. However, the message digest computed is a function of both the input message and the secret key. In this way, only someone with the key can verify the hash value.

When a sender wants to prove to the receiver that the message he sent has not been modified in transit, he would first share a secret key with the receiver. The sender then uses the secret key together with the message itself to generate a message digest from a MAC generation algorithm, and sends the MAC with the message. Once the receiver receives the message, he can also use the secret key to compute a MAC on the received message. If the message received is not altered, the computed MAC should match the received MAC.

2.3.5 Pseudorandom Generator

A pseudorandom number generator (PRNG) is an algorithm that generates a sequence of numbers that approximates a real random number sequence. The pseudorandom numbers play an important role in the theory of cryptography. Careful mathematical analysis is required to place any confidence in a PRNG to generate “sufficiently” random numbers suitable for intended use.

2.3.6 Asymmetric Cryptography

In 1976, Whitfield Diffie and Martin Hellman introduced an entirely new cryptography paradigm [24]. They described public-key cryptography, which uses two different keys—one public and the other private. The keys, although mathematically related, are computationally infeasible to deduce one from the other. This new cryptography paradigm is sometimes referred to as asymmetric cryptography to contrast with the traditional symmetric cryptography.

In public-key cryptography, everyone would have a public key and a private key. To communicate securely under this paradigm, one would first obtain his/her partner's public key, encrypt the message with this key, and send the ciphertext across the network. Once the partner receives the encrypted message, he/she may use his/her private key to decrypt it.

2.3.7 Digital Signature

In public-key cryptography, messages encrypted with the public key can be decrypted with the corresponding private key. Likewise, messages encrypted with the private key can also be decrypted with the corresponding public key. This property can be used to generate the digital counterpart of a written signature. Since a private key is assumed to be known only by its owner, a sender can prove that a message is really originated from him/her by encrypting the message with his/her private key. Anyone with access to the sender's public can thus verify the authenticity of the message by decrypting it.

2.3.8 X.509 Certificates

Although public key cryptography solves the problem of having to share a secret key before communication can begin, the sender still does not have a reliable way of obtaining the public key of the receiver. An attacker could trick the sender into using his/her public key instead of the receiver's and be able to decrypt any message that the sender encrypted with that key. In 1978, Kohnfelder recognized this problem and introduced the concept of using a certificate to convey the public key in his bachelor's thesis entitled “Towards a Practical Public-Key Cryptosystem” [25]. Simply stated, public-key certificates are used to bind an entity's name with the corresponding public key. A trusted third party, called a certificate authority (CA), would be established to verify an entity’s identify, prepare a document containing the entity’s public key, and certify that document by digitally signing it with CA’s private key.

Today, most public-key certificates are based on X.509 certificate format, originally defined in the 1988 recommendation issued by ITU-T. Now, three versions of an X.509 public-key certificate are defined. The original version 1 public-key certificate suffers from inherent inflexibility because this version cannot be extended to support additional attributes. Version 2 public-key certificate offers two optional unique identifier fields for the issuer and the subject and did nothing to correct this shortcoming. Because the demand for these fields was negligible and the same inability to support extensions also applies, version 2 public-key certificate has failed to gain widespread acceptance.

Not surprisingly, the deficiencies associated with the version 1 and version 2 definitions were corrected by the introduction of the version 3 public-key certificates, as specified in the 1997 X.509 Recommendation. Specifically, the addition of optional extensions has given version 3 significant improvements over versions 1 and 2.

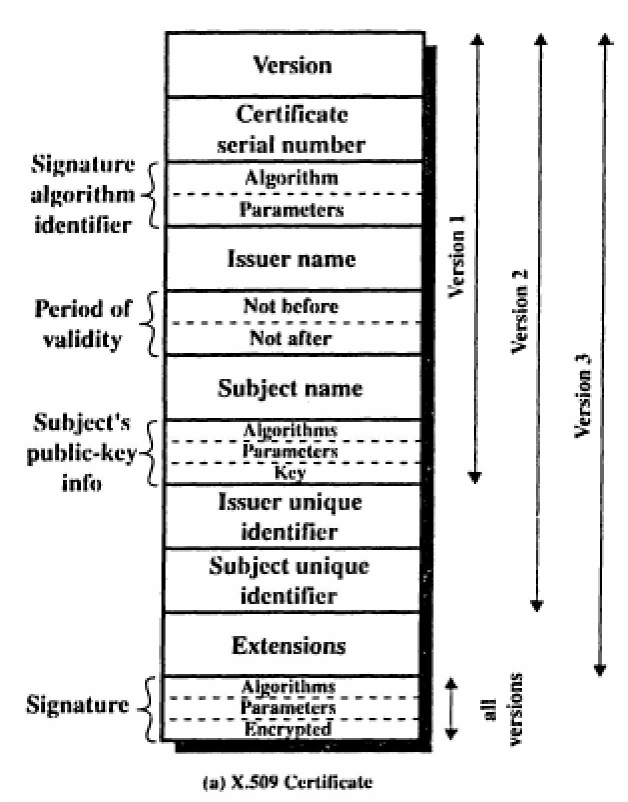


Figure 2-1 X.509 Certificate Format

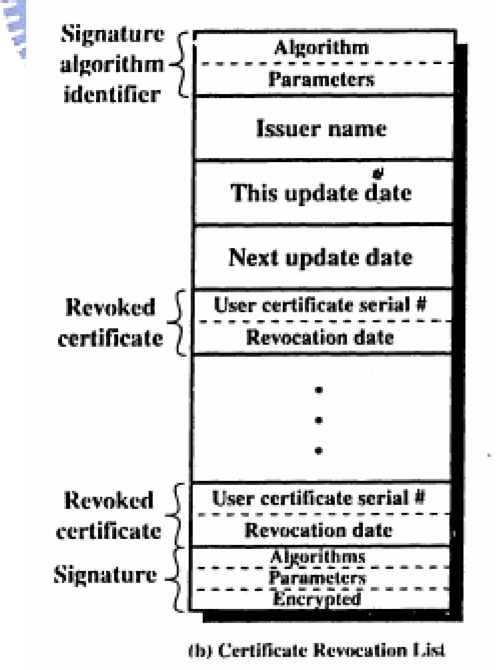


Figure 2-2 X.509 Certificate Revocation List Format

2.3.9 Elliptic Curve Cryptography

Public key cryptography is based on the creation of mathematical entities called trap-door one-way function. These functions are difficult to inverse without certain secret information. The trap-door one-way function can be used to scramble a message in a way that only people with access to the secret information can unscramble. Early public key systems, such as the RSA algorithm, created their trap-door one-way functions based on the difficulty of integer factorization. However, due to recent progress in factoring, RSA public keys must now be thousands of bits long to provide adequate security.

Solving equations of the form $a^b = c$ when a and c are known is easy using logarithms if the numbers involved are real or complex. However, in a large finite group, finding solutions to such equations is quite difficult and is known as the discrete logarithm problem.

An *elliptic curve* is a plane curve defined by an equation of the form

$$y^2 = x^3 + a\,x + b.$$

An operation with properties similar to the integer addition can be defined on the set of points on such a curve with the point at infinity as identify element. If the coordinates x and y are chosen from a large finite field, the solutions form a finite abelian group. The discrete logarithm problem on such elliptic curve groups is believed to be more difficult than the corresponding problem in the multiplicative group of nonzero elements of the underlying finite field. Thus keys in elliptic curve cryptography can be chosen to be much shorter for a comparable level of security.

Unlike other popular public key cryptosystems, no mathematical proof of difficulty has been published for ECC as of 2006. However, American National Standards

Institute (ANSI) has published a digital signature standard based on elliptic curve cryptography in its ANSI X9.62 publication. That standard is also approved in NIST FIPS 186-2 for federal use.

Chapter 3 Advanced Access Content System

3.1 Introduction

On July 14, 2004, eight companies in three key industry sectors joined forces to establish a licensing entity with the responsibility to create and manage a copy protection scheme for entertainment content to be incorporated into the next generation optical storage media. The eight companies are Sony, Toshiba and Matsushita Electronics from the consumer electronics sector; Microsoft, Intel and IBM from the Information Technology sector; and Disney and Warner Brothers from the movie studio sector. The licensing entity they created is forever known as AACS LA, LLC.

After nine months of gestation period, the licensing entity finally comes up with version 0.90 specification on April 14, 2005. The copy protection scheme, Advanced Access Content System (AACS), aims to provide an advanced, robust and renewable method for protecting audiovisual entertainment content, including high-definition content. This version of the specification was published on their website for public review and independent study, and only contained the portion of the design not involving the specifics of any media format. Although AACS LA promised to finalize the specification to version 1.0 by the end of 2005, that time has come and gone. Another ten months has passed since version 0.90 was available, and the AACS specification is updated with version 0.91 at last on February 17, 2006. This chapter will give a brief description of the essential procedures based on the version 0.91 of AACS specification.

3.2 Specification Organization

The specification is organized into several “books”. The books can be categorized into format independent portion and dependent portion. The format independent portion defines the cryptographic procedures that are used to protect audiovisual content stored on pre-recorded and recordable storage media. The format dependent portion then defines additional details for using the system on specific media formats.

Format Independent Books
Introduction and Common Cryptographic Elements Book [1]
Pre-recorded Video Book [2]
Recordable Video Book [3]
Format Specific Books
Blu-ray Disc Pre-recorded Book [4]
Blu-ray Disc Recordable Book [5]
HD DVD and DVD Pre-recorded Book [6]
HD DVD Recordable Book

Table 3-1 AACS Specification Organization

3.3 Common Cryptographic Functions

This section describes the cryptographic functions upon which AACS protection mechanisms are based. The functions are described in isolation, and their specific uses are described later in this chapter.

3.3.1 Advanced Encryption System (AES)

AES is a symmetric block cipher algorithm, as specified in FIPS Publication 197 [9]. The AES algorithm allows three key lengths: 128, 192, and 256 bits and works with messages in blocks of 128 bits each. Since AACS is based exclusively on AES with 128-bit key, this section will only describe this variant.

The AES algorithm proceeds as follows. An input plaintext block consists of 16 bytes, $x_0, \cdots, x_{15}$. The block is initially arranged into a four by four matrix of bytes in a column-wise manner. The matrix is referred to as the State.

$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$		x_0	x_4	x_8	x_{12}
$s_{1,0}$	$s_{1,1}$	$s_{1,2}$	$s_{1,3}$		x_1	x_5	x_9	x_{13}
$s_{2,0}$	$s_{2,1}$	$s_{2,2}$	$s_{2,3}$	←	x_2	x_6	x_{10}	x_{14}
$s_{3,0}$	$s_{3,1}$	$s_{3,2}$	$s_{3,3}$		x_3	x_7	x_{11}	x_{15}

Figure 3-1 AES Input to State Assignment

Then, AES processes the State in a number of rounds. If the key length is 128 bits, then the number of rounds is 10. Each round performs identical tasks with the final round slightly different. The 128-bit secret key is not used directly but is first expanded into a set of round keys.

```
Round(State, ExpandedKey[i])
{
   SubBytes(State);
   ShiftRows(State);
   MixColumns(State);
   AddRoundKey(State, ExpandedKey[i]);
}
```

```
FinalRound(State, ExpandedKey[Nr])
{
   SubBytes(State);
   ShiftRows(State);
   AddRoundKey(State, ExpandedKey[Nr]);
}
```

- AddRoundKey:

  In this transformation, the state is modified by combining it with a round key using the bitwise XOR operation.

- SubBytes:

  In this transformation, each byte of the state is replaced by another byte value. The replacement algorithm is based on advanced mathematics of finite field. However, we could simply implement this operation by hardwiring a lookup table.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	6A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	59	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

- ShiftRows

Each row of the state matrix is cyclically shifted left by different offsets.

a	b	c	d		a	b	c	d
f	g	h	e		e	f	g	h
k	l	i	j	←	i	j	k	l
p	m	n	o		m	n	o	p

- MixColumns

The columns of the state are considered as polynomials over GF(2^8) and multiplied modulo $x^4 + 1$ with a fixed polynomial

$c(x) = 03 \cdot x^3 + 01 \cdot x^2 + 01 \cdot x + 02$.

```
algorithm AES-128E(State, CipherKey)
{
    KeyExpansion(CipherKey, ExpandedKey);
    AddRoundKey(State, ExpandedKey[0]);
    for ($i$ = 1; $i$ < 10; $i$++)
        Round(State, ExpandedKey[$i$]);
    FinalRound(State, ExpandedKey[10]);
}
```

3.3.2 AES in ECB Mode

When managing cryptographic keys, the AES cipher is used in the ECB mode of operation. The AES encryption in ECB mode is represented by the function

$$\text{AES-128E}(k,d)$$

where d is a 128-bit data value to be encrypted, k is a 128-bit key. AES-128E returns the 128-bit encrypted result.

The AES decryption in ECB mode, on the other hand, is represented by the function

$$\text{AES-128D}(k,d)$$

where d is a 128-bit data value to be decrypted, k is a 128-bit key. AES-128D returns the 128-bit decryption result.

3.3.3 AES in CBC Mode

When encrypting and decrypting protected content, the AES cipher is used in the CBC mode of operation. The AES encryption in CBC mode is represented by the function

$$\text{AES-128CBCE}(k,d)$$

where d is a frame of data to be encrypted, k is a 128-bit key. AES-128CBCE returns the encrypted frame.

The AES decryption in CBC mode, on the other hand, is represented by the function

$$\text{AES-128CBCD}(k,d)$$

where d is a frame of data to be decrypted, k is a 128-bit key. AES-128D returns the decryption frame.

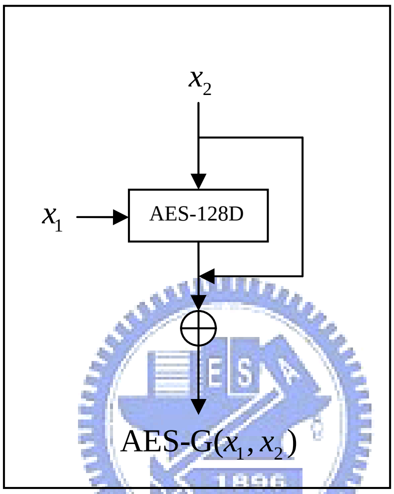
Unless otherwise specified, the Intialization Vector used in CBC encryption and decryption is a constant, $iv_0 = \text{0BA0F8DDFEA61FB3D8DF9F566A050F78}_{16}$.

3.3.4 AES-based One-way Function

AACS defines a cryptographic one-way function based on the AES algorithm. This function is referred to as the AES-based One-way Function, and is represented by

$$\text{AES-G}(x_1, x_2)$$

where x_1 and x_2 are 128-bit input values. $\text{AES-G}(x_1, x_2)$ returns a 128-bit result.



The AES-based One-way Function result is defined by the following formula:

$$\text{AES-G}(x_1, x_2) = \text{AES-128D}(x_1, x_2) \oplus x_2.$$

3.3.5 Secure Hash Algorithm (SHA)

NIST published a Secure Hash Standard in its FIPS Publication 180-2 [7] on August 1, 2002. The standard specifies four secure hash algorithms, SHA-1, SHA-256, SHA-384, and SHA-512, for processing a message to produce a condensed representation called a message digest. When AACS needs to generate or verify digital signatures, it would use SHA-1 algorithm for this purpose.

Each algorithm can be described in two stages: preprocessing and hash computation. Preprocessing involves padding a message, dividing the padded message into *m*-bit blocks, and setting initialization values to be used in the hash computation. The hash computation generates a message schedule from the padded message and uses that schedule, along with functions, constants, and word operations to iteratively generate a series of hash values. The final hash value generated by the hash computation is used to determine the message digest.

The four algorithms differ most significantly in the number of bits of security that are provided for the data being hashed – this is directly related to the message digest length. Additionally, the four algorithms differ in terms of the size of the blocks and words of data that are used during hashing. Table 3-2 presents the basic properties of all four secure hash algorithms.

Algorithm	**Message Size (bits)**	**Block Size (bits)**	**Word Size (bits)**	**Message Digest Size (bits)**
SHA-1	**$< 2^{64}$**	**512**	**32**	**160**
SHA-256	$< 2^{64}$	512	32	256
SHA-384	$< 2^{128}$	1024	64	384
SHA-512	$< 2^{128}$	1024	64	512

Table 3-2 Secure Hash Algorithm Properties

This section only describes SHA-1 algorithm in detail and refers interested readers to the original specification for more information on SHA-256, SHA-384, and SHA-512.

SHA-1 may be used to hash a message of length no longer than 2^{64} bits. During the hash computation, the algorithm uses a message schedule of eighty 32-bit words, labeled $W_0, W_1, \ldots, W_{79}$. The algorithm also requires five working variables, labeled *a*, *b*, *c*, *d*, and *e*, and stores intermediate result into five 32-bit words, denoted $H_0^{(i)}, H_1^{(i)}, \ldots, H_4^{(i)}$. A temporary 32-bit word, *T*, is also used. The following is the pseudo-code for the SHA-1 algorithm.

```
SHA-1($M$)
{
   /* Preprocessing */
   1.  Pad the message, $M$.
   2.  Parse the padded message into $N$ 512-bit blocks, $M^{(1)},\dots,M^{(N)}$.
   3.  Set the initial hash value, $H^{(0)}$

   /* Hash Computation */
   for $i$ = 1 to $N$:
   {
      1.  Prepare a message schedule, $W_t$.
      2.  Initialize the five working variables.
      3.  Update the five working variables.
      4.  Compute the $i$th intermediate hash value, $H^{(i)}$.
   }
   return $H^{(N)}$
}
```

- **Preprocessing**

Preprocessing consists of three steps: padding the message, parsing the padded message into message blocks, and setting the initial hash value, $H^{(0)}$.

The message, *M*, shall be padded to ensure that the padded message is a multiple of 512 bits. Suppose that the message, *M*, is *l* bits in length. The message, *M*, is appended with the bit “1”, followed by *k* zero bits, where $k = \min\left\{x \ge 0 \middle| l + 1 + x \equiv 448 \bmod 512\right\}$, and then appended with the 64-bit block that is equal to the binary representation of the number *l*.

After a message has been padded, it must be parsed into *N* 512-bit blocks before the hash computation can begin. The parsed blocks are denoted by $M^{(1)}, M^{(2)}, \dots, M^{(N)}$. Also, each 512-bit block can be written as a sequence of sixteen 32-bit words. For example, the message block $M^{(i)}$ can be broken up into $M_0^{(i)}, \dots, M_{15}^{(i)}$.

The initial hash value, $H^{(0)}$, must also be set before hash computation commences, and shall consist of the following five 32-bit words:

$$\begin{aligned} H_0^{(0)} &= 67452301 \\ H_1^{(0)} &= \text{efcdab89} \\ H_2^{(0)} &= \text{98badcfe} \\ H_3^{(0)} &= 10325476 \\ H_4^{(0)} &= \text{c3d2e1f0} \end{aligned}$$

- **Hash Computation**

After the preprocessing stage is done, SHA-1 processes each message block in turn to generate the final hash value. For each message block, the processing includes the preparation of a message schedule, the initialization of the working variables, the updating of the working variables and the computation of the intermediate hash value. The final intermediate hash value computed will be the hash value for the original message.

The message schedule consists of eighty 32-bit words, denoted by $W_0, W_1, \ldots, W_{79}$. For the message block $M^{(i)}$, the message schedule is given by

$$W_t = \begin{cases} M_t^{(i)}, & 0 \le t \le 15 \\ ROTL^1(W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16}), & 16 \le t \le 79 \end{cases}$$

The working variables are initialized to the previous iteration's hash value: $a = H_0^{(i-1)}$, $b = H_1^{(i-1)}$, $c = H_2^{(i-1)}$, $d = H_3^{(i-1)}$ and $e = H_4^{(i-1)}$. The working variables are updated through the use of the message schedule, and pre-defined functions and constants.

SHA-1 uses a sequence of logical functions, $f_0, f_1, \ldots, f_{79}$. Each function f_t, where $0 \le t \le 79$, operates on three 32-bit words, *x*, *y*, and *z*, and produces a 32-bit word as output. The function $f_t(x, y, z)$ is defined as follows:

$$f_t(x,y,z) = \begin{cases} Ch(x,y,z) = (x \wedge y) \oplus (\neg x \wedge z) & 0 \le t \le 19 \\ Parity(x,y,z) = x \oplus y \oplus z & 20 \le t \le 39 \\ Maj(x,y,z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z) & 40 \le t \le 59 \\ Parity(x,y,z) = x \oplus y \oplus z & 60 \le t \le 79 \end{cases}$$

SHA-1 also uses a sequence of eighty 32-bit constants, $K_0, K_1, \ldots, K_{79}$, given by

$$K_t = \begin{cases} 5a827999 & 0 \le t \le 19 \\ 6ed9eba1 & 20 \le t \le 39 \\ 8f1bbcdc & 40 \le t \le 59 \\ ca62c1d6 & 60 \le t \le 79 \end{cases}$$

The working variables are updated by the following loop:

```
for $i$ = 0 to 79,
{
    $T = ROTL^5(a) + f_t(b,c,d) + e + K_t + W_t$
    $e = d$
    $d = c$
    $c = ROTL^{30}(b)$
    $b = a$
    $a = T$
}
```

Finally, the i^{th} intermediate hash value is computed: $H_0^{(i)} = a + H_0^{(i-1)}$, $H_1^{(i)} = b + H_1^{(i-1)}$, $H_2^{(i)} = c + H_2^{(i-1)}$, $H_3^{(i)} = d + H_3^{(i-1)}$, and $H_4^{(i)} = e + H_4^{(i-1)}$. The N^{th} intermediate hash value, $H^{(N)}$, is returned as the 160-bit hash value.

3.3.6 AES-based Hashing Function

In certain calculations involving keys, data may need to be processed to produce a condensed representation. To this end, AACS defines a hash function based on the AES algorithm. This procedure, referred to as the AES-based Hashing Function, is represented by

AES-H(M)

where *M* is input data of arbitrary length. AES-H(*M*) returns the corresponding 128-bit hash value.

Before the hash value can be computed, the data to be hashed is padded using the standard SHA-1 method as described in section 3.3.5. The padded message is then parsed into blocks of length 128-bit each. AES-H processes these blocks sequentially to compute the message digest. The algorithm can be described by the following pseudo-code.

```
algorithm AES-H($M$)
{
   /* Preprocessing */
   Pad the message, $M$.
   Parse the padded message into $N$ 128-bit blocks, $M^{(1)},\ldots,M^{(N)}$.
   Set the initial hash value,
      $H^{(0)} \leftarrow$ 2DC2DF39420321D0CEF1FE2374029D95$_{16}$;

   /* Hash Computation */
   for $i$ = 1 to $N$ do
   {
      Compute the $i^{\text{th}}$ intermediate hash value $H^{(i)}$
      $H^{(i)} \leftarrow$ AES-G($M^{(i)}, H^{(i-1)}$)
   }
   return $H^{(N)}$
}
```

3.3.7 Cipher-based MAC (CMAC)

AACS requires a message authentication code algorithm to protect the integrity of information. In 2005, NIST described a method of incorporating a symmetric block cipher to create message authentication code in its Special Publication 800-38b. The MAC function they described is thus called Cipher-based MAC, CMAC for short. CMAC depends on the choice of an underlying symmetric block cipher. The CMAC key is the same as the block cipher key. AACS chooses AES with key length of 128 bits as the underlying cipher algorithm.

The input to the MAC generation function is a bit string called the message, denoted M, with the bit length $Mlen$. The output of the MAC generation function is a bit string called the MAC, denoted T. The length of T, denoted $Tlen$, is 128-bit as stated in the specification.

The block cipher key is used to derive two additional secret subkeys, denoted K_1 and K_2. The subkeys has the same length as the block cipher key. During the subkey generation process, a bit string, denoted R_b, is initialized according to the cipher block length. In particular, for block length $b = 128$, $R_{128} = 0^{120}10000111$. In general, R_b is the binary representation of a certain irreducible polynomial of degree *b*. The subkeys are generation with the following steps:

```
algorithm $GenSubkey(K)$
1.  $L \leftarrow CIPH_K(0^b)$;
2.  $K_1 \leftarrow L << 1$;
3.  if $\mathrm{MSB}_1(L) = 1$ then
        $K_1 \leftarrow K_1 \oplus R_b$;
4.  $K_2 \leftarrow K_1 << 1$;
5.  if $\mathrm{MSB}_1(L) = 1$ then
        $K_2 \leftarrow K_2 \oplus R_b$;
6.  return $K_1, K_2$
```

As for any MAC algorithm, an authorized party applies the MAC generation process to produce a MAC for the data authentication purpose. Subsequently, any authorized party can apply the same MAC generation to the data and compare the computed MAC with the received MAC. The verification process fails if a mismatch is detected.

The specification of CMAC algorithm for MAC generation is as follows:

```
algorithm $CMAC(k, M)$
1.  $K_1, K_2 \leftarrow GenSubkey(k)$
```

```
2.  if $Mlen = 0$ then
        $n \leftarrow 1$;
    else
        $n \leftarrow \lceil Mlen/b \rceil$;
3.  Parse $M$ into $n$ blocks such that
        $M = M_1 \| M_2 \| \cdots \| M_{n-1} \| M_n^*$
    where $M_1, M_2, \ldots, M_{n-1}$ are complete blocks.
4.  if $M_n^*$ is a complete block then
        $M_n \leftarrow K_1 \oplus M_n^*$;
    else
        $M_n \leftarrow K_2 \oplus (M_n^* \| 10^j)$ where $j \leftarrow nb - Mlen - 1$;
5.  $C_0 \leftarrow 0^b$;
6.  for $i \leftarrow 1$ to $n$ do
        $C_i \leftarrow CIPH_K(C_{i-1} \oplus M_i)$;
7.  $T \leftarrow \mathrm{MSB}_{Tlen}(C_n)$;
8.  return $T$;
```

3.3.8 Random Number Generator

Random/pseudorandom number generators are incorporated by AACS to generate values such as cryptographic keys. Unless stated otherwise, one or more of the following generators shall be used:

1. Pseudorandom number generator based on a design described in ANSI X9.31.
2. Pseudorandom number generators defined in FIPS PUB 186-2 (+Change Notice).
3. Random or pseudorandom number generator of equal or higher quality as measured by the tests described in NIST Special Publication 800-22 when using the default parameters and other recommendations provided therein.

3.3.9 Digital Signature Algorithm

All digital signatures in AACS utilize the ECDSA algorithm, which is based on elliptic curve cryptography, and is defined in ANSI X9.62 publication. NIST also approves in FIPS PUB 186-2 (+Change Notice) that ECDSA is suitable to be used for digital signature generation and verification.

The following table shows the elliptic curve and the underlying finite field that shall be used in conjunction with ECDSA.

ECC Parameter	Value
p	900812823637587646514106462588455890498729007071
a	-3
b	366394034647231750324370400222002566844354703832
Base Point (*G*)	264865613959729647018113670854605162895977008838
	51841075954883162510413392745168936296187808697
Order of Base Point (*r*)	9008128236375876465141065555566573588779770753047

Table 3-3 AACS Parameters for ECDSA

The elliptic curve, $y^2 = x^3 + ax + b$, with the values of *a* and *b* taken from the table, and the associated operation are defined over GF(*p*). Each digital signer shall process two keys. The public key, K_{pub}, which is a point on the given elliptic curve, and the private key, K_{priv}, which is a scalar value satisfying $0 < K_{priv} < r$, shall be chosen to satisfy the equation:

$$K_{pub} = K_{priv} G \, .$$

The ECDSA digital signature generation function is represented by the following function in AACS specification:

$$S = \text{AACS-Sign}(K_{priv}, D)$$

where K_{priv} is the signer’s private key, *D* is the data to be signed and can be of any length, and *S* is the 40-byte resulting signature.

The ECDSA digital signature verification function is represented by the following function in AACS specification:

$$S = \text{AACS-Verify}(K_{pub}, S, D)$$

where K_{pub} is the signer’s public key, *S* is the received signature data, *D* is the data whose signature is to be verified.

3.4 Key Management

Under AACS scheme, audiovisual content is stored on optical storage media in encrypted form. The content decryption key would also need to be stored on the same optical disc in such a way that only through a specialized procedure performed by authorized devices can retrieve. A large part of that specialized procedure is governed by a data structure called the Media Key Block (MKB). Based on the subset-difference tree method, MKB provides system renewability in the form of device revocation.

Each compliant device is given a set of secret Device Keys by AACS LA when manufactured. Those keys are used to decrypt one or more elements of a MKB, in order to extract a secret Media Key (K_m). MKB embeds within itself device revocation information so that compromised devices would not be successful in extracting the secret Media Key.

3.4.1 Subset-Difference Tree (NNL Tree)

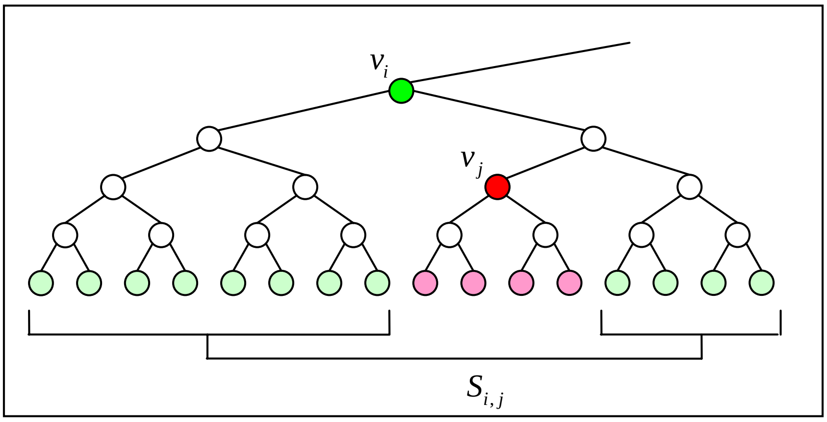
In June 2002, Naor, Naor and Lotspiech [26] presented a Subset-Cover framework that summarizes several previously known broadcast encryption solutions in their “Revocation and Tracing Schemes for Stateless Receivers”. They also suggested two new schemes within that framework; the second scheme, the subset difference tree approach, has been adapted in AACS specification as the basis for the MKB data structure. This section shall attempt to rewrite the algorithm presented in [26] in light of the description of MKB in AACS [1].

Described in terms of content distributed on optical media, the subset difference tree revocation approach can be used to solve the following problem. With $\mathbb{N}$ the set of all devices and $\mathbb{R} \subset \mathbb{N}$ the devices that are revoked, the replicator can distribute specially packaged discs to the devices such that any device $u \in \mathbb{N} \setminus \mathbb{R}$ can recover the content, while even a coalition of all devices in $\mathbb{R}$ cannot.

The revocation system consists of three parts: (1) Device keys assignment stage, which assigns secret keys to devices that allow them to decrypt; (2) Content distribution stage, which formulates device revocation information given the set of revoked devices, $\mathbb{R}$; (3) Content decryption stage, which only permit non-revoked devices to decrypt the encrypted content given their set of secret device keys.

Within a Subset-Cover framework, the set of devices are organized into a list of subsets $S_1, S_2, \ldots, S_w$. Each subset S_j is associated with a processing key L_j such that every member $u \in S_j$ should be able to derive L_j from its secret information. Given a revoked set $\mathbb{R}$, the remaining, non-revoked devices are partitioned into disjoint set $S_{i_1}, S_{i_2}, \ldots, S_{i_m}$ such that $\mathbb{N} \setminus \mathbb{R} = \bigcup_{i=1}^{m} S_{i_j}$. The media key, which is used to hide the content decryption key, is then encrypted *m* times using $L_{i_1}, L_{i_2}, \ldots, L_{i_m}$ and stored in the media key block data structure.

The subset-difference tree approach views the set of all devices as the leaves of a complete binary tree. The collection of subsets corresponds to subsets of the form $G_1 \setminus G_2$ with $G_2 \subset G_1$. The two groups G_1, G_2 represent leaves in two full binary subtrees. Therefore, $S_{i,j} = G_1 \setminus G_2$ can be represented by a pair of nodes (v_i, v_j) where v_i is an ancestor of v_j.



Processing Key Assignment

AACS defines a cryptographic pseudorandom sequence generator that triples the length of a seed value. The generator, AES-G3, is defined by

$$\text{AES-G3}(k)=G_L(k)\,||\,G_M(k)\,||\,G_R(k)$$

where $G_L(k)=\text{AES-G}(k,s_0)$, $G_M(k)=\text{AES-G}(k,s_0+1)$, $G_R(k)=\text{AES-G}(k,s_0+2)$, and $s_0=\text{7B103C5DCB08C4E51A27B01799053BD9}_{16}$.

AACS maintains an enormous complete binary tree and assigns a secret key to each node of the tree. Consider a complete subtree T_i rooted at v_i, and let the secret key associated with the root v_i be k_i. Then, the processing key $L_{i,j}$ for the subset $S_{i,j}$ is derived through the following recursive labeling process.

To begin, the node v_i is labeled $\text{LABEL}_i=k_i$. If a node has been labeled *k*, its left and right children will be labeled $G_L(k)$ and $G_R(k)$, respectively. Let $\text{LABEL}_{i,j}$ be the label of v_j derived in this way starting from having the label of v_i equal to LABEL_i. Then, the processing key $L_{i,j}$ for the set $S_{i,j}$ will be $G_M(\text{LABEL}_{i,j})$.

Device Keys Assignment

Suppose a device correspond to a leaf node *u* of the tree T_i rooted at v_i. Consider the path from u to v_i and let $v_{i_1},v_{i_2},\ldots,v_{i_k}$ be the nodes not on the path, but whose parent is on the path. It should be noted that if $u \in S_{i,j}$, then v_j is not on the path from u to v_i, and therefore v_j must be a descendent of one of $v_{i_1}, v_{i_2}, \ldots, v_{i_k}$. Hence, if we assign $\mathrm{LABEL}_{i,i_1}, \mathrm{LABEL}_{i,i_2}, \ldots, \mathrm{LABEL}_{i,i_k}$ to the device, it will be able to derive $L_{i,j}$.

3.4.2 Media Key Block

The Media Key Block (MKB) enables system renewability. Every AACS-protected content is accompanied with a MKB that contains the media key K_m that only compliant devices, each using their secret Devices Keys, can successfully retrieve. Given the known compromised devices $\mathbb{R}$, AACS LA determines the partition $S_{i_1,j_1}, S_{i_2,j_2}, \ldots, S_{i_m,j_m}$ of compliant devices. The media key is then encrypted *m* times with $L_{i_1,j_1}, L_{i_2,j_2}, \ldots, L_{i_m,j_m}$ and stored inside the MKB structure.

If a device is not revoked, it should find its membership in one of the set $S_{i_1,j_1}, S_{i_2,j_2}, \ldots, S_{i_m,j_m}$. Once, the device determines which subset $S_{i,j}$ it belongs to, it can then use its assigned Device Keys to calculate the required decrypting key $L_{i,j}$ to retrieve K_m. Otherwise, the device would not find its place among the collection of sets, and cannot proceed to calculate the media key. The device key representation and MKB layout are described in length in AACS specification and shall not be reproduced in this thesis.

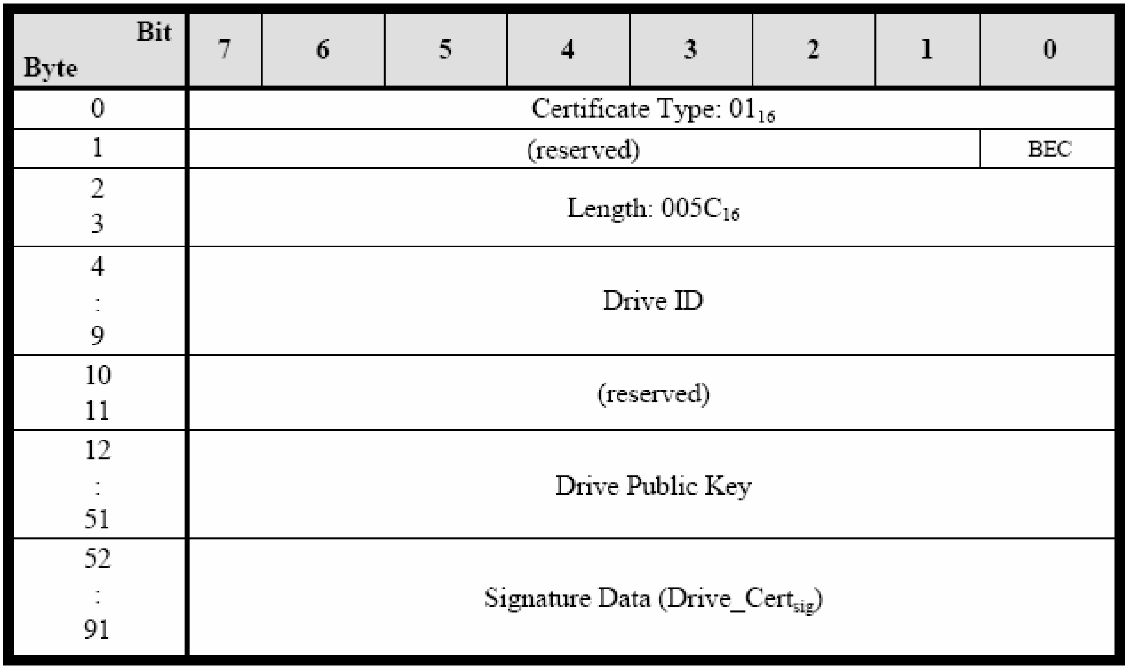
3.5 Drive-Host Authentication

AACS specification is applicable to PC-based systems. In such a system, a drive and PC host act together as the Recording Device and/or Playback Device. A Drive-Host Interface exists between the drive and the host, and the communication flowing through that interface can be compromised. Therefore, the PC host will need to read and verify information received from the drive, and also ensure information is securely written to the media by the drive. Consequently, AACS defines a two-way Drive-Host authentication procedure for such a configuration.

3.5.1 Drive and Host Certificates

In order to perform the required drive-host authentication, a drive manufacture must request a certificate to be issued from AACS LA, and include that certificate within a corresponding drive. The certificate is used to hold the drive's public key. The same thing can also be said about compliant PC hosts.

Each drive and PC host has a private key and a public key. They must also have access to AACS LA's public key, $AACS\_LA_{pub}$. Figure 4-1 shows the format of the Drive Certificate.



Bit Byte	7	6	5	4	3	2	1	0
0	Certificate Type: 01_{16}							
1	(reserved)							BEC
2 3	Length: $005C_{16}$							
4 : 9	Drive ID							
10 11	(reserved)							
12 : 51	Drive Public Key							
52 : 91	Signature Data ($Drive\_Cert_{sig}$)							

Figure 3-2 Drive Certificate

3.5.2 Drive Authentication Algorithm

By the drive authentication, the drive and the PC host verify each counterpart is AACS compliant and has valid certificate signed by the AACS LA. They also verify each other has the ability to sign and verify signatures, and is not revoked by checking the drive revocation list (DRL) and the host revocation list (HRL). When the drive authentication is successful, the drive and the PC host will have a shared Bus Key (BK), and can proceed to next steps. Figure 3-3 illustrates the drive authentication algorithm as defined in the AACS specification. The algorithm is referred to as AACS-Auth.

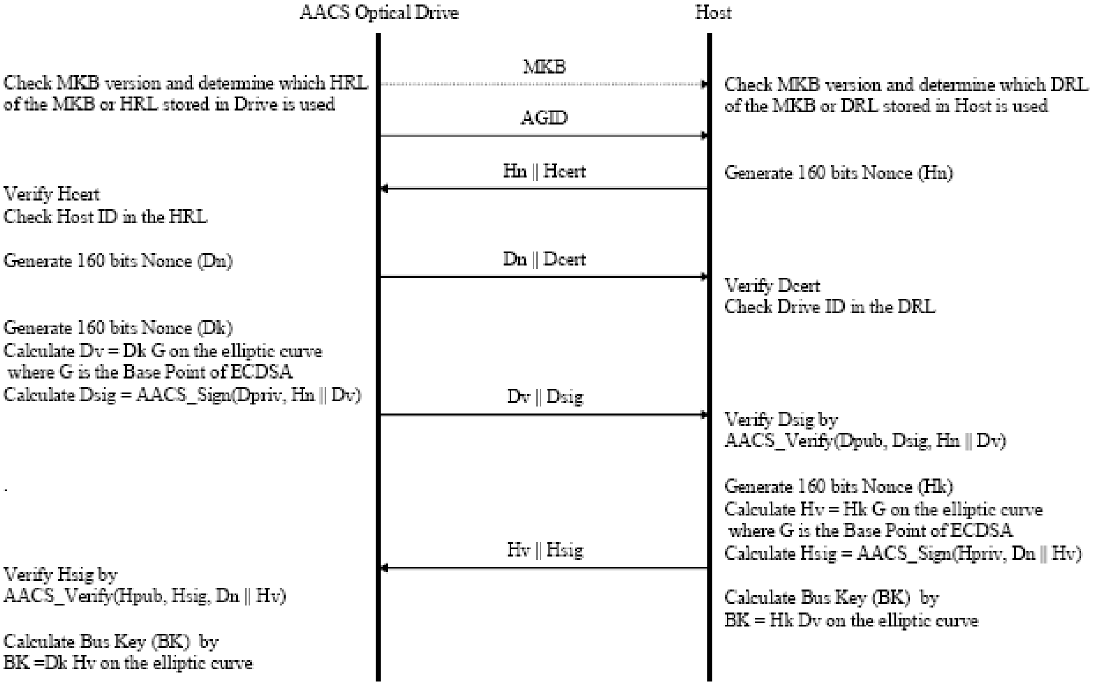

AACS Optical Drive
Host
MKB
Check MKB version and determine which HRL of the MKB or HRL stored in Drive is used
Check MKB version and determine which DRL of the MKB or DRL stored in Host is used
AGID
Hn || Hcert
Generate 160 bits Nonce (Hn)
Verify Hcert
Check Host ID in the HRL
Generate 160 bits Nonce (Dn)
Dn || Dcert
Verify Dcert
Check Drive ID in the DRL
Generate 160 bits Nonce (Dk)
Calculate Dv = Dk G on the elliptic curve
where G is the Base Point of ECDSA
Calculate Dsig = AACS_Sign(Dpriv, Hn || Dv)
Dv || Dsig
Verify Dsig by
AACS_Verify(Dpub, Dsig, Hn || Dv)
Generate 160 bits Nonce (Hk)
Calculate Hv = Hk G on the elliptic curve
where G is the Base Point of ECDSA
Calculate Hsig = AACS_Sign(Hpriv, Dn || Hv)
Hv || Hsig
Verify Hsig by
AACS_Verify(Hpub, Hsig, Dn || Hv)
Calculate Bus Key (BK) by
BK =Dk Hv on the elliptic curve
Calculate Bus Key (BK) by
BK = Hk Dv on the elliptic curve

Figure 3-3 Drive Authentication Algorithm for AACS [1]

3.5.3 Protocol for Transferring Information

Several identifiers are used in AACS defined procedures. The Volume Identifier is an identifier that is unique to each individual item of content, the Pre-recorded Media Serial Number identified each pre-recorded media, and the Media Identifier identifies each recordable media. These identification numbers must be securely transferred between the drive and the host, and similar procedures are performed for this purpose. Figure 3-4 demonstrates the procedure in the case of the Volume Identifier.

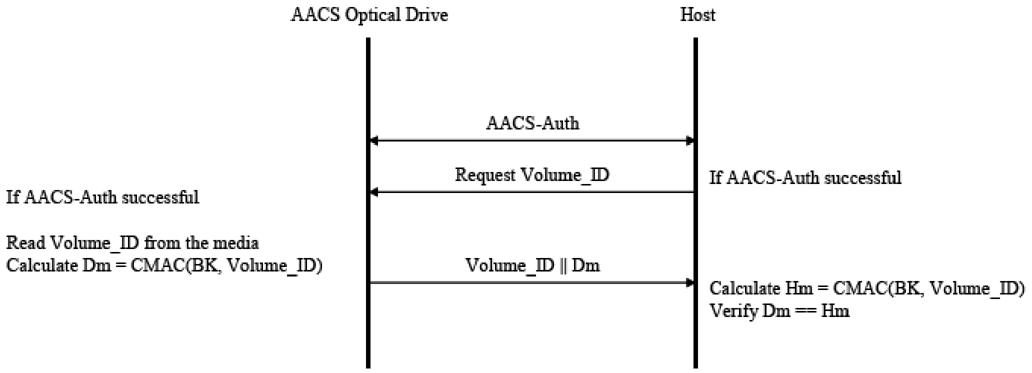


Figure 3-4 Protocol for Transferring Volume Identifier [1]

3.5.4 Protocol for Updating the Protected Area and Associated on Recordable Media

The Protected Area is used to store and restore the Binding Nonce created by the drive on a recordable media. Two protocols are defined for reading from and writing to the Protected Area Data. When a host wants to request the drive to update the Binding Nonce, the host should read and cache the current value of the Binding Nonce, request the drive to commit a new Binding Nonce, and then verify a new value of the Binding Nonce is indeed written to the Protected Area of the recordable disc.

The two-way authentication procedure, AACS-Auth, is initiated at the beginning of the two protocols. If the authentication is successful, the host can then continue to send his request to read or write. The drive processes the request by reading the old value of the Binding Nonce, or generating a new value of the Binding Nonce to be written. In either cases, the Binding Nonce is returned to the host along with a message digest to protect the integrity of the transmitted message.

Figure 3-5 and Figure 3-6 illustrates the procedures and data flow for the two protocols.

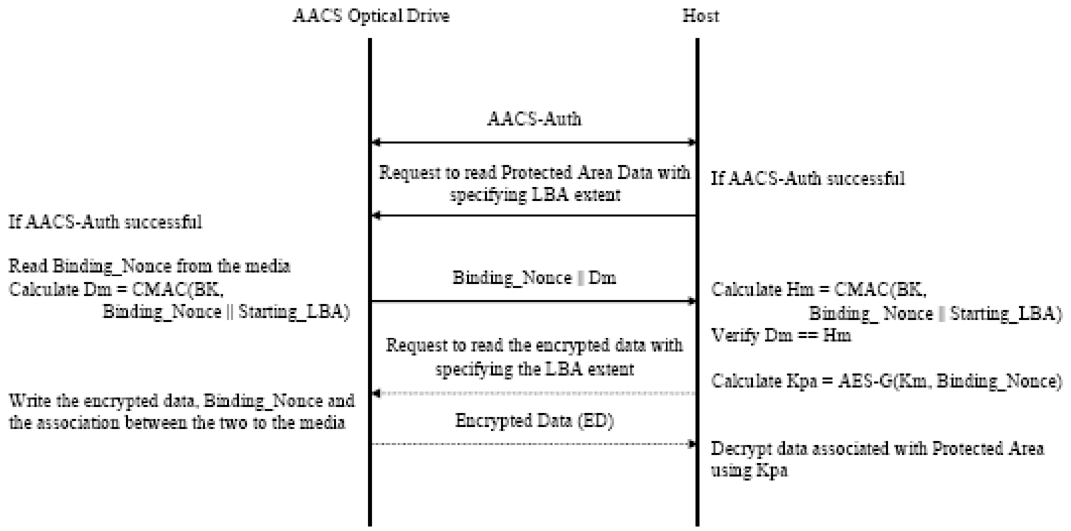

AACS Optical Drive
Host
AACS-Auth
Request to read Protected Area Data with specifying LBA extent
If AACS-Auth successful
If AACS-Auth successful
Read Binding_Nonce from the media
Calculate Dm = CMAC(BK, Binding_Nonce || Starting_LBA)
Binding_Nonce || Dm
Calculate Hm = CMAC(BK, Binding_ Nonce || Starting_LBA)
Verify Dm == Hm
Request to read the encrypted data with specifying the LBA extent
Calculate Kpa = AES-G(Km, Binding_Nonce)
Write the encrypted data, Binding_Nonce and the association between the two to the media
Encrypted Data (ED)
Decrypt data associated with Protected Area using Kpa

Figure 3-5 Protocol Flow for Reading Protected Area Data [1]

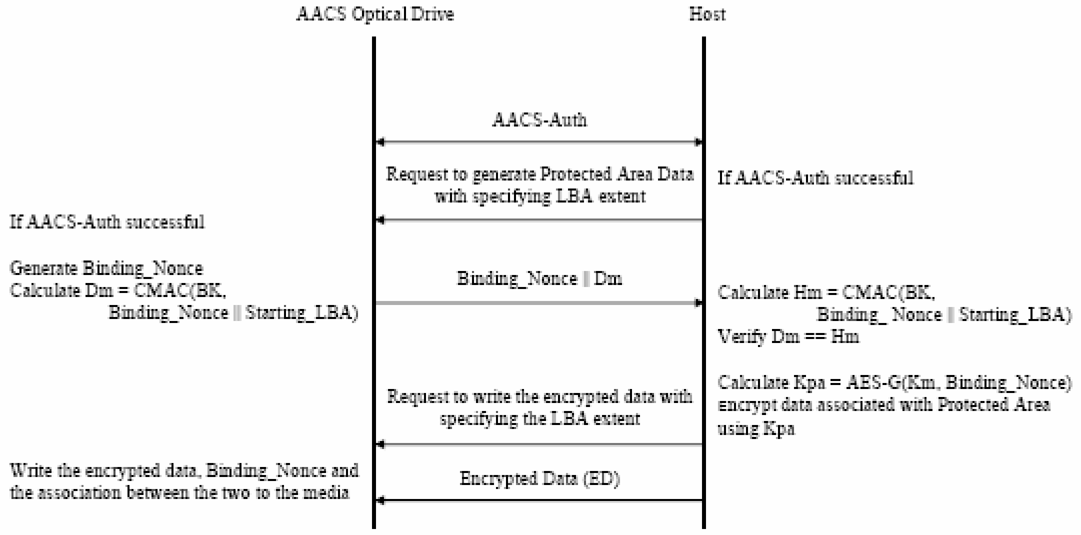

AACS Optical Drive
Host
AACS-Auth
Request to generate Protected Area Data with specifying LBA extent
If AACS-Auth successful
If AACS-Auth successful
Generate Binding_Nonce
Calculate Dm = CMAC(BK, Binding_Nonce || Starting_LBA)
Binding_Nonce || Dm
Calculate Hm = CMAC(BK, Binding_ Nonce || Starting_LBA)
Verify Dm == Hm
Calculate Kpa = AES-G(Km, Binding_Nonce)
encrypt data associated with Protected Area using Kpa
Request to write the encrypted data with specifying the LBA extent
Write the encrypted data, Binding_Nonce and the association between the two to the media
Encrypted Data (ED)

Figure 3-6 Protocol Flow for Writing Protected Area Data [1]

Chapter 4 Proposed Scheme

In the AACS Specification, several certificate types have been defined. Content certificates allow a playback device to verify the integrity of the content before the content is played. Host certificates and drive certificates are used in PC environments to ensure the legitimacy of the drive and the application program on the computer. Through the authorization granted by a remote server, AACS protected pre-recorded content could be copied onto an alternative media. This process is called Managed Copy. Before accepting the authorization, the device making the copy must first ensure that the remote server is authentic and not some machine set up by an attacker. MCS certificates are created just for this purpose. In their design, AACS LA will act as the only certificate issuing facility and each certificate type has its own structure.

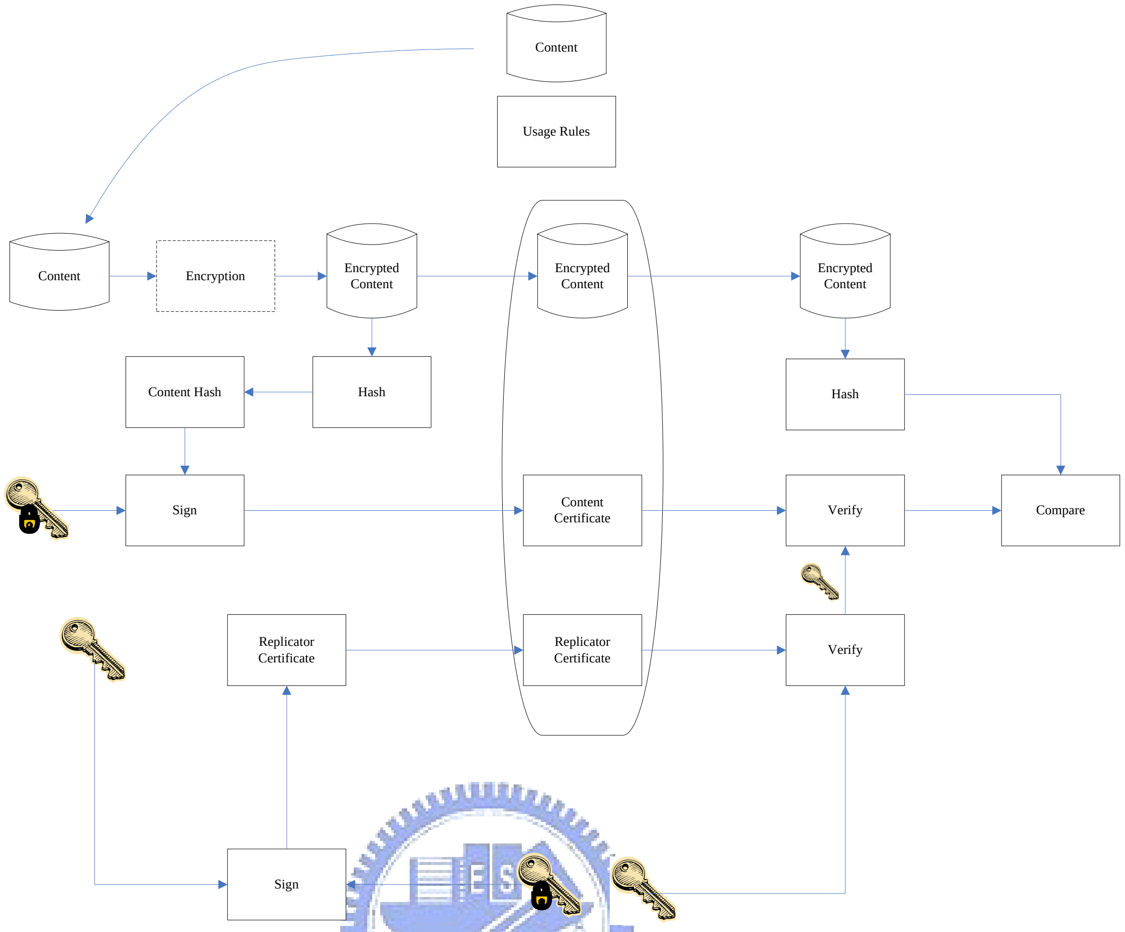
In this chapter, we propose a scheme that redefines AACS certificates using X.509 version 3 certificate format. This scheme permits the AACS LA to become the root certificate authority that could appoint other facilities to specialize in one type of certification. However, this modification requires the usage of each certificate type to be changed as well. In the following sections, we will take each certificate type in turn and discuss what changes must be incorporated into the original AACS design.

4.1 Content Certificate

In the original design, before the replicator can mass produce the AACS protected content, it would need to create a content hash table to be signed by AACS LA before the content certificate can be issued. Although the replicator only needs to apply for the content certificate once, the procedure nonetheless disrupt the normal production cycle.

In our proposed scheme, AACS LA would delegate the generation of content certificates to the replicator. This is accomplished by making the replicator an subordinate certificate authority of AACS LA. The content certificate generation phase is essentially the same. However, after the hash table digests are calculated and collected into a certificate to be signed, the replicator will sign the certificate using its private key instead. In addition to the content certificate, the replicator would also place the replicator certificate onto the disc so that a licensed player could verify the signature on the content certificate.

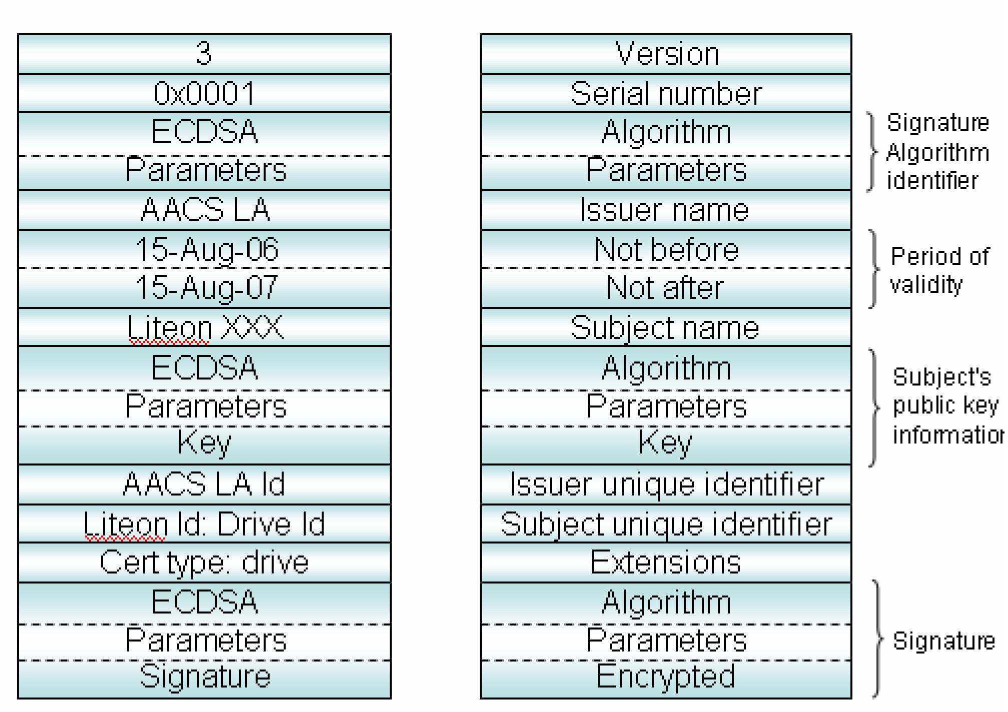
When a licensed player needs to verify the authenticity and integrity of the content certificate, it would notice that the certificate was signed by a replicator. The player would need to check that the replicator is trusted by AACS LA by verifying the replicator certificate stored on disc using AACS LA's public key. After the player determines that the replicator is trustworthy, the player can then fetch the replicator's public key from the certificate and use the key to verify the authenticity and integrity of the content certificate. If the content certificate is not revoked, the player would calculate some number of content hash and match the results with the ones stored in the content certificate. If all verification steps succeed, the player will proceed with the playback.



4.2 Drive Certificate

In the PC environment, the optical disc drive and the PC host are combined to form a single playback/recording device. Since some sensitive information must be transmitted through the unsecure PC/Drive Interface, we must be careful to guard against the occurrence of fabricated data. Hence, a drive certificate is issued by AACS LA to prove that the information provided by the drive actually comes from the inserted optical disc.

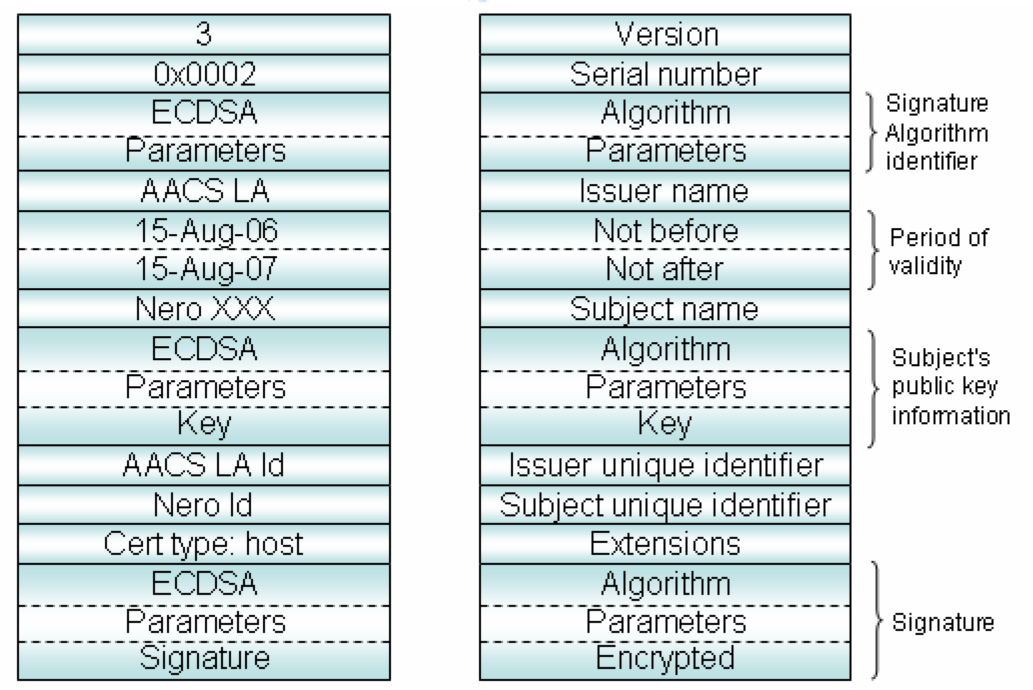
The major information stored in a drive certificate is the drive id and the corresponding public key. AACS will then append a signature using its private key. It is quite straightforward to adapt the certificate into X.509 certificate format. An example is shown in the following figure.



3	Version	
0x0001	Serial number	
ECDSA	Algorithm	Signature Algorithm identifier
Parameters	Parameters	
AACS LA	Issuer name	
15-Aug-06	Not before	Period of validity
15-Aug-07	Not after	
Liteon XXX	Subject name	
ECDSA	Algorithm	Subject's public key informatior
Parameters	Parameters	
Key	Key	
AACS LA Id	Issuer unique identifier	
Liteon Id: Drive Id	Subject unique identifier	
Cert type: drive	Extensions	
ECDSA	Algorithm	Signature
Parameters	Parameters	
Signature	Encrypted	

4.3 Host Certificate

In the PC environment, the PC host also needs to be authenticated. A host certificate is provided by AACS LA for this purpose. Since a host certificate is structurally similar to a drive certificate, the change toward X.509 format is also similar.



3	Version	
0x0002	Serial number	
ECDSA	Algorithm	Signature Algorithm identifier
Parameters	Parameters	
AACS LA	Issuer name	
15-Aug-06	Not before	Period of validity
15-Aug-07	Not after	
Nero XXX	Subject name	
ECDSA	Algorithm	Subject's public key information
Parameters	Parameters	
Key	Key	
AACS LA Id	Issuer unique identifier	
Nero Id	Subject unique identifier	
Cert type: host	Extensions	
ECDSA	Algorithm	Signature
Parameters	Parameters	
Signature	Encrypted	

Chapter 5　Conclusion

At the time of this writing, AACS specification is still under development. We could anticipate major alterations once a weakness is discovered. Those alterations may include new certificate types to provide authentication and integrity services for various entities in the AACS universe. Instead of defining new structures for each additional certificate types, we could use the extension facility provided by X.509 version 3 certificate type to define private data fields required as the proposed scheme demonstrated.

Additionally, the original design places too much burden upon the licensing agency since it would need to manage the certificate issuance and revocation for many different participants. Our proposed scheme allows AACS LA to designate other facility to specialize in managing a certain type of certificates.

Reference

[1] AACS LA, LLC. “Advanced Access Content System v0.91: Introduction and Common Cryptographic Elements”, http://www.aacsla.org/specification.

[2] AACS LA, LLC. “Advanced Access Content System v0.91: Pre-recorded Video Book”, http://www.aacsla.org/specification.

[3] AACS LA, LLC. “Advanced Access Content System v0.91: Recordable Video Book”, http://www.aacsla.org/specification.

[4] AACS LA, LLC. “Advanced Access Content System v0.91: Blu-ray Disc Pre-recorded Book”, http://www.aacsla.org/specification.

[5] AACS LA, LLC. “Advanced Access Content System v0.91: Blu-ray Disc Recordable Book”, http://www.aacsla.org/specification.

[6] AACS LA, LLC. “Advanced Access Content System v0.91: HD DVD and DVD Pre-recorded Book”, http://www.aacsla.org/specifications.

[7] AACS LA, LLC. “Advanced Access Content System v0.91: HD DVD Recordable Book”, http://www.aacsla.org/specifications.

[8] FIPS 180-2, “Secure Hash Standard”, National Institute of Standards and Technology, 2002.

[9] FIPS 197. “Advanced Encryption Standard (AES)”, National Institute of Standards and Technology, 2001.

[10] NIST SP 800-38a. “Recommendation for Block Cipher Modes of Operation – Methods and Techniques”, National Institute of Standards and Technology, 2001.

[11] DVD Forum. http://www.dvdforum.com.

[12] Kazaa software version 3.0 [Online]. Available: http://www.kazaa.com.

[13] BitTorrent software version 4.20.4 [Online]. Available: http://www.bittorrent.com/download.html.

[14] eDonkey software v1.4 [Online]. Available: http://www.edonkey2000.com.

[15] Gnuttela [Online]. Available: http://www.gnutella.com.

[16] Content Scramble System [Online]. Available: http://www.dvdcca.org

[17] B. Schneier. Applied Cryptography, 2nd Ed., John Wiley & Sons, Inc., 1996.

[18] J. Vacca. ed. Public Key Infrastructure: Building Trusted Applications and Web Services, CRC Press, 2004.

[19] J. Bloom et al. “Copy protection for DVD video,” in Proc. IEEE, vol. 87, Jul. 1999, pp. 1267–1276.

[20] J. Taylor. DVD Demystified. New York: McGraw Hill, 1998.

[21] P. Biddle et al., “The Darknet and the Future of Content Distribution,” in Proc. ACM Workshop Digital Rights Management 2002. [Online] Available: http://crypto.stanford.edu/DRM2002/darknet5.doc.

[22] D. Touretzky. (2000) Gallery of CSS Descramblers. [Online]. Available: http://www.cs.cmu.edu/~dst/DeCSS/Gallery.

[23] R. Merkel, “Secrecy, Authentication, and Public Key Systems,” Ph.D. dissertation, Stanford University, 1979.

[24] W. Diffie and M. Hellman. “New Directions in Cryptography,” IEEE Transactions on Information Theory, v. IT-22, n. 6, Nov 1976, pp. 644-654.

[25] L. Kohnfelder. “Toward a Practical Public-Key Cryptosystem,” B. Sc Thesis, MIT Department of Electrical Engineering, 1978.

[26] D. Naor, M. Naor, and J. Lotspiech. “Revocation and Tracing Schemes for Stateless Receivers”, Advances in Cryptology – CRYPTO 2001, Springer-Verlag Inc. LNCS 2139, 2001, pp. 41-62.

[27] Eskicioglu, M. Ahmet and E. Delp. “An Overview of Multimedia Content Protection in Consumer Electronics Devices”, Signal Processing: Image Communication 16, pp. 681-699, Elsevier publishers, 2001.

[28] B. Schneier. “The Futility of Digital Copy Prevention”, Crypto-Gram Newsletter, May 15, 2001. Available: http://www.schneier.com/crypto-gram-0105.html.