Cracking Advanced Access Content Protection (April 2007)

Daniel Dmytriw, Neale Genereux, Michael Gujral, Abhishek Valaboju

Abstract—**Digital rights management (DRM) has long been a popular issue in the realm of content copy protection and its application to the film industry. As an analysis, this paper explores the recent efforts to circumvent the DRM scheme used on HD DVD media. CSS is described to motivate the methods used in cracking the current AACS standard. A detailed explanation of the attack conducted by the authors is presented, and possible countermeasures are discussed.**

Index Terms—**Advanced Access Content System, AACS, Digital Rights Management, DRM, High Definition Digital Video Disc, HD DVD**

I. Introduction

DIGITAL rights management (DRM) has long been a hot button issue in the realm of content copy protection especially in its application to the film industry. The Advanced Access Content System (AACS) is the content protection standard that has been adopted by the entertainment industry to manage content on the next generation of optical media discs. These discs, which include HD DVD and Blu-Ray, have been created to replace the smaller-capacity DVD format, and to allow a richer multimedia experience through high definition audio and video. Having learned from the ease with which DVD's content protection system was circumvented, the creators of AACS designed a robust AES-based encryption system with several security features.

Despite all of the effort that went in to creating this robust content protection system, a crack has recently surfaced on the internet only a few weeks after the first AACS protected media became available. Instead of attacking the AES-based encryption, the attack attempts to obtain the encryption keys themselves from commercial media player applications. Our project is aimed at reproducing and analyzing this attack on the AACS system, analyzing the severity of the problem, and suggesting ways in which these security holes can be patched within the current system.

Manuscript received April 13, 2007.
D. Dmytriw (e-mail: dmytriw@telus.net).
N. Genereux (e-mail: incarnaterw@gmail.com).
M. Gujral (e-mail: michael.gujral@gmail.com).
A. Valaboju (e-mail: abhishek.rv@gmail.com).

II. High Definition Digital Video Disc

High Definition Digital Video Disc (HD DVD) is one of the two new disc formats designed to store high definition (HD) video; The other is Blu-Ray. For the purposes of this paper and the crack it describes, we limit our scope to HD DVD, although everything presented is theoretically applicable to Blu-Ray as well. An HD DVD has the same dimensions as a regular DVD, but it represents a significant evolution in terms of the storage capacity and in turn application. The extra space is put to use with very high quality 1080p resolution video and 7.1 digital surround sound which yield an improved user experience [1].

III. Digital Rights Management

For the purposes of this paper, DRM technology serves to protect copyrighted material stored on digital media from being copied.

The emergence of media sharing applications such as peer-to-peer networks and BitTorrent has forced digital content distributors to rethink what buying a CD, DVD, or HD DVD actually means. If the user can easily share any media they own, then sales of media will decline since not every consumer needs to buy the media to access it. However, if the user simply licenses media from the distributor and is prevented from copying it by the license agreement, then each consumer must still purchase the media to view it. This view of digital media has caused content providers to move toward a licensing scheme for their products. DRM technology is their means of enforcing the "no-copy" clause within the end user license agreement [2].

One early attempt at including DRM on digital media was the Content Scramble System (CSS) used on DVDs. This system is described in the next section, while its successor, AACS, is described in section V.

IV. Content Scramble System

One of the first DRM systems came in the form CSS for DVDs in 1996. This system was used to prevent the copying of DVDs as well as to prevent the playback of DVDs in players that were not officially licensed by the DVD Copy Control Association (DVD CCA). The encryption used was a relatively weak 40-bit stream cipher which was proven to be susceptible to brute force attacks. The weakness was due not only to US

government crypto-export regulations but also the trusted client problem. In general, a trusted client problem occurs when systems are sold to "trusted" clients that the seller doesn't actually trust. [3]

Several keys are needed to decrypt DVDs using CSS. Keys are licensed by the DVD CCA to both DVD media manufacturers as well as DVD player manufacturers. Authentication keys are used to allow the DVD player to authenticate with a CSS decryption module. Title keys are needed to descramble video segments called titles. Sometimes there are multiple titles per DVD with each requiring its own title key. Title keys are obtained by using the DVD disc key which is obtained by the player using the player key. DVD players can have one of about 400 player keys which are assigned by the DVD CCA. [4]

The exact inner workings of the CSS algorithm were originally a closely guarded proprietary secret. However, in late 1999, a piece of reverse engineered code called DeCSS was posted anonymously on the internet. This code demonstrated how to decrypt a player key, disc key and title key. Although the code did nothing on its own other than revealing the details of the CSS algorithm, it was very quickly integrated into many open source video players and spread over the internet.

The release of the algorithm also revealed a major weakness. Since there were a total of 2^{40} possible disc keys used to make a given disk key hash, an offline brute force attack could be run using the key-hash checking portion of the algorithm to find a working disc key. It was demonstrated that the complexity of reversing a key hash was on the order of 2^{25} and any given hash could be reversed to determine its disc key in a matter of seconds using a 450MHz CPU [5]. After a valid disc key is found, it is very easy to test the disc key to obtain one of the possible 409 player keys. Once one player key and disc key are found, title keys can be decrypted and used to decrypt the content on the DVD.

Since the DVD market had already committed to the CSS standard, it could not move to a better solution until a new DVD technology was developed. It is important to note that CSS was the predecessor to AACS and that even though it was a proprietary security system, it was eventually reverse engineered and susceptible to brute force attacks.

V. Advanced Access Content System

AACS was designed to supercede CSS and fix the fundamental security flaws that had been previously implemented in the CSS standard. Firstly, AACS was implemented using an open standard in accordance with the "open design" principle of computer security. Secondly, AACS was based on the 128-bit AES standard which is not currently susceptible to brute force attacks. Finally, AACS was designed to have many layers of security to prevent a single breach from compromising the entire system. Even though AACS was developed to fix all the issues that had been previously been exposed within CSS, as will be seen in later sections of this report, it was ultimately broken.

The actual implementation of AACS is split across two main objects: the pre-recorded media and the playback device. The main components of the AACS system are shown in the figure below.

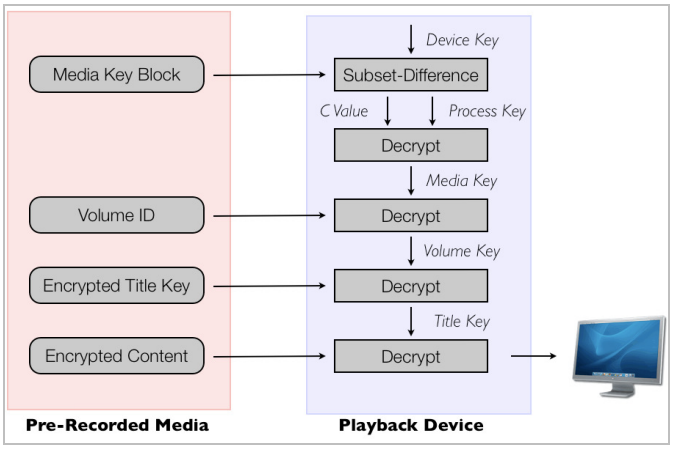

Fig. 1: AACS Architecture

The various components on the pre-corded media are combined with various parts and keys within the playback device to decrypt the HD DVD content. The decryption process starts with the device key which is provided by the playback device itself. This device key is then able to combine with the media key block (MKB) to produce a C-value and a processing key. The C-value is an encrypted media key which can be decrypted with the processing. The media key block is simply a file on the HD DVD that contains 513 C-values, along with some addition data used to find the entry matching the player's device key. The subset difference algorithm is used to determine which C-value corresponds to the given device key [6]. This algorithm is part of the open AACS standard, and is not discussed in this report as it is completely circumvented by the attack described. Combining the processing key and C-value yields the media key [6].

The next phase of the decryption involves the volume ID which is unique to each HD DVD title. Combining the volume ID and media key yields the volume key which ultimately leads to the decryption of the encrypted content [6]. With the volume key it is trivial to retrieve the title key for each media file on the disc using the open AACS standard. While this system was designed with the problems of CSS in mind, it was still cracked as is shown in the next section.

VI. Cracking AACS

In order to circumvent the AACS security system and gain access to volume keys, two approaches were used. The first was a simple attack which surfaced in December of 2006, but was not reproducible. The second was a more complex attack first proposed in February of 2007. This is the attack which was reproduced and refined.

A. Hardware

The HD DVD drive used in each attack was the Microsoft Xbox 360 HD DVD player connected to a standard PC running Windows XP. No other hardware was required.

B. Approach #1: Direct Volume Key Extraction

On December 26, 2006 a member of the doom9 forums (forum.doom9.org) named muslix64 posted a tool called BackupHDDVD which could be used to decrypt and backup HD DVD movies [7]. This utility simply implemented part of the open AACS standard, and required a configuration file with the volume key for each disc the user wished to decrypt in order to function. Posted along with the tool was a video of muslix64 decrypting an HD DVD, and a screenshot of a configuration file containing a few volume keys. This immediately raised the question: how did muslix64 acquire these volume keys?

On the 27th of December, muslix64 answered the question in a vague sense, writing that the keys were in memory while the movies were being played by a commercial HD DVD player software package [7]. Presumably, if one knew what to look for then the keys could be extracted. This was the basis for approach #1 to cracking AACS.

Two commercial software packages were used to attempt this attack: CyberLink PowerDVD 7 and InterVideo WinDVD 8 (with HD support). The posts made by muslix64 did not specify which player was used in the original attack, so the two most popular players were tried. Using keys already available online [8], and the WinHex memory monitor, HD DVD's were played using each software package and the program memory was scanned for the presence of the volume key. The hope was to find known keys within the memory dump, discern a pattern in their location, and then create a tool which automated their extraction. However, neither PowerDVD nor WinDVD had the keys present in memory during playback. This meant that muslix64, and the other posters who were submitting volume keys to the message boards were using some other software package. It also meant that this vulnerability was specific to a single player and not attributable to the AACS standard.

Further research revealed that the player being used with this attack was most likely the Japanese version of InterVideo WinDVD 8 [9]. This product is no longer available through online sales on InterVideo's Japanese website and cannot be obtained without having a boxed copy shipped from Japan. Without the player containing this security hole it was impossible to obtain decrypted volume keys from a software player's memory. Therefore, this attack was not feasible.

C. Approach #2: Generating Volume Keys from Volume ID and MKB

Since it was not possible to retrieve the volume keys directly from software players, the next approach was to generate the volume keys using the open AACS standard. As discussed in the previous section, a volume key may be generated through a combination of a volume ID and a media key. Another member of the doom9 forums named arnezami proposed a method of acquiring these components on the 5th of February, 2007 [10]. This section details the retrieval of both of these components based on this work.

D. Volume ID Interception

When a commercial HD DVD playing software package begins playing a movie, it must first retrieve the volume ID from the media it wishes to play. To do this it sends a request to the hardware player which includes its own device key, and the player sends a reply with a volume ID. According to the AACS standard, this exchange should be secure and encrypted [11]. However, it was discovered that the portion of the transmission traveling over the USB connection is in fact unencrypted. Therefore, if the format of the reply message is known, then the volume ID can be intercepted during this initial handshaking process.

Since the HD DVD player used in this attack was USB based, a simple USB sniffing program called usbsnoop (http://benoit.papillault.free.fr/usbsnoop/) was enough to intercept these transmissions. In order to ensure the initial transmissions were intercepted, the snooper was installed on all of the HD DVD drive's objects and the drive with a disc inserted was plugged into the USB port. At this point, the USB log would be populated while the commercial HD DVD software (PowerDVD 7 in this case) performed an autoplay of the disc. Once the movie began to play, the log would definitely contain the volume ID, so the snooping was terminated.

The results of this process was a USB log file about 200 to 300 MB in size, containing at least one instance of the volume ID and several transmissions of no consequence to the attack. In order to find the volume ID within this much data, a search was performed based on the AACS standard's format for volume identifiers. Each volume identifier starts with a one byte type field, which is always 40_{16} for HD DVDs. This is followed by a single reserved byte, 12 bytes of unique volume identification data, and then two final reserved bytes [11]. Each reserved byte is currently set to 00_{16}. Thus, the format searched for within the log was:

40 00 xx xx xx xx xx xx xx xx xx xx xx xx 00 00_{16}

After conducting this search on a few USB log files, it was determined that the volume ID was being sent in a 36 byte message which started with the value 00 22 00 00_{16}. Thus the volume ID could be found by searching for messages starting with the value 00 22 00 00 40 00_{16}. This was enough information to make the search return one unique result per log file in almost all cases. In those cases with more than one result, it was very easy to determine which was the volume ID based on message length.

E. Media Key Generation

The second object required to generate a volume key is the media key. Since key extraction from commercial player memory was not possible given the available software, this key also had to be generated. According to the AACS specifications, this key is acquired by a commercial player by using a device key to process the media key block (MKB) stored on the HD DVD media. As mentioned in the section describing AACS, the device key is used to generate a processing key and a single C-value/index pair using the subset

difference algorithm. The doom9 poster arnezami has succeeded in extracting a single processing key which has been used on all HD DVD's produced to date [12]. This processing key was used to perform a brute force attack on the MKB file in order to generate the media key. It was hoped that arnezami's process key extraction method could be reproduced, but it required the same piece of Japanese software necessary for the direct extraction of volume keys. Since this software is no longer available, the published processing key was used without further investigation.

With the processing key, a brute force search of the MKB file can be performed to extract the correct C-value and generate the media key. The MKB contains 513 16 byte C-values, each with a unique 4 byte UV number which is used as part of the subset difference algorithm [6]. It also contains an entry named the verify media key record, which is a single 16 byte value used to verify prospective media keys. When the verify media key record is decrypted using a valid media key, the first 8 bytes of the result are 01 23 45 67 89 AB CD EF_{16}. By applying the process key to each of the 512 C-values (combined with their corresponding UV numbers as described by the AACS standard [6]), and checking the result against the verify media key record, the correct media key can be found within 513 iterations. It is interesting to note that in every case tested so far, the correct media key has been found in the very first entry in the MKB.

F. *Volume Key Generation*

With the volume ID intercepted using USB snooping, and the media key generated using an HD DVD's MKB file and the processing posted by arnezami, the volume key can easily be generated. According to the AACS standard, it is a simple matter of decrypting the volume ID using the media key, and then XORing the result with the volume ID [6]. A simple C++ tool was created which takes the MKB file and the snooped volume ID as inputs, and outputs a volume key. This method was used to successfully generate a volume key for every movie attempted.

The timeline below describes key milestones in the cracking of AACS:

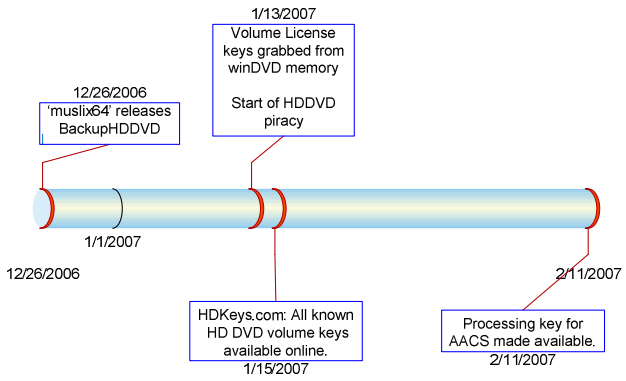

Fig. 2. AACS Crack Timeline

G. *Possible Countermeasures*

The AACS Licensing Association (AACS LA) has several options available to it to counteract the attack described in the previous section. First, there is a key revocation system built into the AACS standard. By removing specific device keys from the subset difference tree in the MKB files of all subsequent media that is manufactured, the AACS can prevent a particular player from decrypting new HD DVD discs. This will make the first attack described obsolete if the problem player is no longer able to retrieve the volume keys of new media. In addition, if the processing key is changed for future media, and the device key is revoked for the player used to retrieve it, a new method of extracting the processing key will have to be implemented before the second attack will work again.

In order to prevent the interception of volume ID's, which, as described above, are an integral part of generating volume keys, the AACS could start requiring that the transmission of volume ID's from player hardware to software be encrypted over the USB connection. In addition, current volume ID's follow a very predictable pattern, and could likely be guessed if they could not be intercepted. They often include the release date of the media or the title of the film. Sometimes they simply have a long string of 20_{16}'s at the end of the ID. By randomizing these values, the AACS LA could increase the difficulty in guessing volume ID's by several orders of magnitude.

Finally, as a complete but somewhat drastic solution, the AACS LA could stop granting device keys to software players. This would limit the use of HD DVD technology on computers to data storage and the playback of unprotected media. All AACS protected media would have to be played on a hardware player connected to a compatible HD TV. Although this solution would require hackers to extract keys from hardware devices instead of software, and thus make this extraction process much more difficult, it is not overly attractive. Consumers would likely be upset and confused if their computer's HD DVD drive could not play HD DVD's.

VII. The Future of Digital Media

AACS, while not currently a trivial DRM system to crack, is by no means perfect. The countermeasures prescribed above and others serve to make the technology more resistant to hacks, but at the cost of usability. Early adopters of HD DVDs will not be pleased to have a player, hardware or software, that at one time played HD DVDs but at some point cease to be able to do so.

DRM technology used on media which is viewable and distributed to millions of people is inherently breakable. Since the content providers are giving consumers both the media they are trying to protect and the means to access it, malicious users have all the tools necessary to retrieve the data on these discs. The only real solution in this situation is to make the protection mechanism so complicated that attackers cannot circumvent it before the next generation of media is released. However, the technology needs to balance an adequate and acceptable level of usability to propagate consumer psychological acceptability while keeping production costs within a reasonable range. Due to the quantities of discs produced, the smallest cost increase if only to minutely improve end-to-end system security in both player and media

would represent many millions of dollars invested over the life of HD DVD technology.

Indeed, history has a tendency to repeat itself. CSS was cracked. AACS has been cracked. All signs point to future technologies being cracked. This paper concludes with the position that DRM itself is a misapplication of what would otherwise be a solid, well designed piece of security technology.

REFERENCES

[1] Tech Specs [Online]. Available: http://www.thelookandsoundofperfect.com/

[2] Digital Rights Management, *Wikipedia*. [Online]. Available: http://en.wikipedia.org/wiki/Digital_Rights_Management

[3] Content Scramble System, *Wikipedia*. [Online]. Available: http://en.wikipedia.org/wiki/Content_Scramble_System

[4] The DVD Content Scrambling System Explained. [Online]. Available: http://dvd-copy.blogspot.com/2005/08/dvd-content-scrambling-system.html

[5] Cryptanalysis of Contents Scrambling System, *Frank A. Stevenson (frank@funcom.com)* http://www.dvd-copy.com/news/cryptanalysis_of_contents_scrambling_system.htm

[6] Intel Corporation, International Business Machines Corporation, Matsushita Electric Industrial Co., Ltd., Microsoft Corporation, Sony Corporation, Toshiba Corporation, The Walt Disney Company, Warner Bros. (2006, Feb 17). Advanced Access Content System (AACS): Introduction and Common Cryptographic Elements. [Online]. Available: http://www.aacsla.com/specifications/specs091/AACS_Spec_Common_0.91.pdf

[7] muslix64. (2006, Dec 27). BackupHDDVD, a tool to decrypt AACS protected movies. *doom9 forums.* [Online]. Available: http://forum.doom9.org/showthread.php?t=119871

[8] (2007, April 11). HDKeys.com. [Online]. Available: http://www.hdkeys.com/

[9] (2007, April 11). Decrypted Title Keys and Volume Unique Key for BackupHDDVD to Bypass HD-DVD AACS DRM. *My Digital Life*. [Online]. Available: http://www.mydigitallife.info/2007/01/15/decrypted-title-keys-and-volume-unique-key-for-backuphddvd-to-bypass-hd-dvd-aacs-drm/

[10] arnezami. (2007, Feb 5). Processing Key, Media Key and Volume ID found!!!. *doom9 forums.* [Online]. Available: http://forum.doom9.org/showthread.php?t=121866

[11] Intel Corporation, International Business Machines Corporation, Matsushita Electric Industrial Co., Ltd., Microsoft Corporation, Sony Corporation, Toshiba Corporation, The Walt Disney Company, Warner Bros. (2006, Aug 15). Advanced Access Content System (AACS): HD DVD and DVD Pre-recorded Book. [Online]. Available: http://www.aacsla.com/specifications/AACS_Spec_HD_DVD_and_DVD_Prerecorded_0_912.pdf

[12] arnezami. (2007, Feb 11). Processing Key, Media Key and Volume ID found!!!. *doom9 forums.* [Online]. Available: http://forum.doom9.org/showthread.php?p=952954#post952954