Digital Rights Management

Rune Hammersland and Jonas Strømstad
Students at NISlab

July 10, 2008

Abstract

Digital Rights Management, or DRM, is a term used for hardware and software that enables a content provider to place restrictions on the content they provide. These restrictions can be "simple" restrictions, like the inability to play/display the content in/on unsupported hardware or software. They can also be more advanced, and provide the publisher with an opportunity to "rent out" a digital copy, and at the same time feel confident that the copy will be unusable once the rent period is over.

This paper will briefly discuss the advantages DRM gives the publishers, and the disadvantages it poses for the consumers. We will also describe a couple of different DRM schemes, most of them applied to audio and video content.

Digital Rights Management may sound like a good idea seen from the viewpoint of a publisher, but it might not be as good as it sounds ...

Contents

1 An Introduction to Digital (Rights|Restrictions) Management

The term DRM generally stands for Digital Rights Management. However, critics of the concept usually refer to it as Digital Restrictions Management, and it's easy to see that this really is a more fitting term. The companies who'd like to apply DRM technology to their content will of course refer to it as Rights Management. After all, the technology is "protecting" their rights, or at least what they believe are their rights. Users of the content will most often see DRM as an annoyance.

DRM technology is used for the protection of digital content. Most often music, movies and ebooks, but it's also being applied to other documents of text (think: documents from your office applications). As DRM is mostly about *denying* users the rights to perform a certain action, it really is mostly about applying *restrictions*. Hence: Digital Restrictions Management.

2 Looking At It From Two Different Angles

2.1 Why Does DRM Sound So Compelling For the Content Providers?

From the viewpoint of the content providers, DRM can sound very compelling. Content providers have been struggeling a long time with people who are selling illegal copies of their content. DRM might seem like a guarantee against "piracy", but people with some technical insight will know that DRM is just a minor bump in the road. An inconvenience that can be overcome. Still, as long as the inconvenience connected with circumventing the DRM protection is big enough, it might be enough hinder the majority from making illegal copies.

Another reason that makes the use of DRM seem like a good choice, is that it can give the content providers an opportunity to provide "subscription based access" to music on the Internet. This can be done by offering content with "expiration dates". If a customer downloads a song, he/she can be given a key to decrypt the song that will only be valid for a month. If the customer doesn't continue the subscription for the service, he/she will not recieve a new key to decrypt the songs downloaded from the service.

When the content provider feels it's safe to make their content available in a web-based store, they will also make a huge reduction in costs associated with logistics, like shipping the content to different parts of the country/world. Sadly enough, many content providers feels that illegal reproduction is a bigger problem than it really is, so the main reason they are using for deploying DRM technologies are to prevent this. DRM will probably never stop the big crooks, as they will always find a way to work around the DRM restrictions (even though it might mean going through the analog hole). Regular users will most often buy the content they like and can afford.

2.2 Why Does DRM Harm the Consumer?

DRM techniques will usually prevent the customer's right to "fair use". If you buy a CD in a regular music store, you are allowed to sell it to someone else at a later time. You will also have the right to use portions of the tracks in, say, education (note: not the whole CD). It is also your right to take a backup copy of the CD, in case you should break it, and you also have the right to make a copy to play in your car, or at your cabin[1]. Fair use also covers a lot of other rights[2], and many of these are being made hard to achieve if your content is restricted by DRM. As mentioned earlier, DRM is harming the regular consumer more than it is harming the real copyright offenders. The people profiting from copyright infringement will always find a way around the restrictions.

Another problem, that most content publishers neglect to see, is that the people who are profiting

[1] Making copies is at least allowed according to the laws in Norway, but if we're not mistaken, this is also the case in other countries

[2] Actually Norwegian law states that you have the right to make copies to your family and "close" friends, but EU regulations are threatning to remove this part of the law.

from sales of illegal copies, are the people who are willing to make the biggest effort to circumvent the techniques. The ordinary users, who might want to take private copies, will then have greater problems with what is concidered fair use of the content they have legally purchased. Some executives also claims that when a person illegaly downloads copyrighted material, they would have bought the same material if it wasn't available for download illegally. Many people downloads ie. songs to get an impression of a new album, or widen their musical horizon. Some of these people would indeed buy the album if they like it, but many people cannot afford buying music that often, and if the album was not available for download illegally, they wouldn't have bought a legal copy either (and maybe never developed appreciation for the artist's work, which in turn would make them buy later works).

3 Implementations

Here, we'll take a short look at a couple of applications of DRM in real life.

3.1 CSS and Region Codes

CSS is short for Content Scrambling System, and is used to prevent customers from making (illegal) copies of their (legally purchased) content, in this case DVD movies. It uses a proprietary stream cipher of 40 bits, and the reason for this is the export laws conserning cryptography in the United States of America. Until 1996 (when the export laws were relaxed), there was a restriction on the strength of encryption keys, stating that they could not have a length greater than 40 bits [3]. DVD player manufacturers are given one of about 400 "player keys" to embed in their players. The player key is used to decrypt the "disc key" found on a DVD disc, and the disc key is then used to decrypt the "title keys". Lastly the title keys are used to decrypt the data for the title, where a title can be the complete movie, a trailer, or something similar. There is also an authentication process going on, between the DVD drive and the "CSS Decryption Module". The stream cipher used in CSS was reverse engineered in 1999, by two unnamed germans, and the Norwegian Jon Lech Johansen, who released a program called DeCSS. There is also a "Gallery of CSS Descramblers" on the Internet [Tou00] which is maintained by Dr. David S. Touretzky, an expert witness in a court case about CSS in New York, August 2000. There has also been written an unknown number of programs similar to DeCSS, just to prove how weak the protection was. The stream cipher used has, through cryptanalysis [Ste], proven to be much weaker than it potentially could be.

> CSS was designed with a 40 bit keylength to comply with US government export regulation, and as such it easily compromised through brute force attacks (such are the intentions of export control). Moreover the 40 bits have not been put to good use, as the ciphers succumb to attacks with much lower computational work than which is permitted in the export control rules. Whether CSS is a serious cryptographic cipher is debatable. It has been clearly demonstrated that its strength does not match the keylength. If the cipher was intended to get security by remaining secret, this is yet another testament to the fact that security through obscurity is an unworkable principle.
>
> – Frank A. Stevenson

Region Codes are used to limit the countries a DVD can be viewed in. The argument for this restriction, is to be able to have different release dates for different parts of the world (i.e. a movie can be released on DVD in the United States of America, while it is still being shown on cinemas in Europe). The world is divided into nine regions, of wich six is used on movies sold in stores [Inf]:

0. Playable in all regions.
1. Bermuda, Canada, United States and U.S. territories
2. The Middle East, Western Europe, Central Europe, Egypt, French overseas territories, Greenland, Japan, Lesotho, South Africa and Swaziland

[3] US Export of Cryptography

3. Southeast Asia, Hong Kong, Macau, South Korea and Taiwan
4. Australia, New Zealand, Central America, the Caribbean, Mexico, Oceania and South America
5. The rest of Africa, Former Soviet Union, the Indian subcontinent, Mongolia and North Korea
6. Mainland China
7. Reserved for future use (found in use on protected screener copies of MPAA-related DVDs, and "media-copies" of pre-releases in Asia)
8. International venues such as aircraft, cruise ships, etc.

Many electrical stores will offer you to remove the region lock on a given DVD player (by setting the region of the player to region 0). Some countries see the region coding as a violation of the free trade agreements, and as such, DVD players sold in these countries are all set to region 0. Another big problem with the region lock, is that even though it is originally used to make it possible to ship DVDs of a movie still playing in cinemas in the rest of the world, reissues will often contain region codes as well. Putting a region restriction on a twenty year old movie / TV show will go against the original goal of region coding, and will require people with DVD players set to a specific region to make sure the DVDs they buy has the correct region code. Most countries have no laws against removing the region lock.

3.2 CopyControl

CopyControl is the generic name of several copy protection mechanisms on audio CDs, and it's goal is the same as CSS - to prevent the customer from making copies of their content. As audio CDs with CopyControl does not comply with the CD standard, the discs will not have the familiar CDDA-logo. To make sure disgruntled customers won't complain about being unable to make copies of their CopyControlled CDs, the CopyControlled discs are labeled with the CopyControl logo.

Several methods can be used. Multisession information can be included, which will "hide" the audio tracks from most CD-ROM drives, as they assume the disc contains data. The data section of the disc can also contain DRM versions of the tracks on the disc, as well as a proprietary music player, capable of playing the DRM'ed songs. In some cases the data section can also contain DRM software which needs to be installed on your computer before you are able to play the tracks (depending on the company, the software may be autoinstalled; see section 3.3). To reduce the sound quality of a ripped copy, the error correction codes can also be intentionally corrupted.

There are, however, several ways around the CopyControl mechanisms, and that is probably the reason this restriction mechanism is dropping in use. CD-R/RW drives and DVD-R/RW drives will usually have no problems identifying the two (or more) sessions on the disc, and will gladly let you play the content on the audio session (regular CD-ROM and DVD-ROM drives will usually not). Also, the proprietary player placed on the data section is usually targeted for the Microsoft Windows operating system, as well is the autoplay feature they rely on for the player to start. If you are running a different operating system, the player will probably not autostart, and if it does, the chances are small that it will run on your system. Last, but not least, there is always the analog hole. By recording the sound played on the computer's sound card, you will reduce the recording speed, but if you have a digital output and input on the sound card, there will be little, or no loss of quality.

3.3 Sony/BMG's rootkit-variants

In 2005, Sony and BMG used two methods called Extended Copy Protection (XCP) and MediaMax CD-3 on roughly 100 CD's [4]. The software was included on the CDs, and installed when a user tried to play the CD in the CD-ROM on their computer. The software behaves much the same way as a rootkit does: it is installed without the user's knowledge, it patches the system kernel to make sure ordinary system tools will not list it as a running process, and intercepts calls to the CD-ROM

[4] CD's with XCP or MediaMax

drive to make sure only the proprietary player installed from the CD is allowed access to the drive. The player allows the user to make a limited number of copies of the CD, and allows the user to transfer the tracks to certain MP3 players (the Apple iPod is a notable exception). The XCP software also includes a "Plug and Play Device Manager" to monitor all the programs being run on the computer. XCP only targets the Microsoft Windows operating system, while MediaMax also targets Apple's Mac OS X (even though the premissions of Mac OS X helps the users to become aware of what's happening).

One of the biggest problems with the XCP DRM technique is that it opens up holes that malicious software can exploit. The patch to the kernel that hides XCP from process listings is programmed to hide processes that start with the string `$sys$`. After being preassured by bad media coverage, Sony/BMG released software to remove XCP (or rather, made the company responsible for the development of XCP release the uninstall software). The uninstaller also contains a number of critique worthy concerns:

- To get to the download site, you have to fill out a form, leaving your e-mail. Then you will recieve a personalized link to a second form in an e-mail, before being able to download the uninstaller. Sony's privacy statment states that your e-mail may be used to send you promotions, and may be given to affiliates.
- To download the uninstaller, you have to install an ActiveX component that renders your computer open for attack from other sites (by allowing them to run software on the computer without restrictions).
- The uninstaller only removes parts of the XCP software.

However, Sony have now included a direct link to the uninstall software[5], so these concerns have been taken care of. Note also that to prevent the XCP software to launch when you insert a XCP protected CD, you can hold down the "Shift" key to prevent AutoPlay to kick in. There is also a registry setting to disable AutoPlay, which can easily be set through TweakUI which can be obtained from Microsoft[6], or through Group Policy Editor which comes with Microsoft Windows XP Professional.

There are also other problems with XCP. If you uninstall the software from your computer, it will be installed next time you try to play a CD with XCP. If you want to exchange the CD with a version not containing XCP, you may file a claim for settlement from Sony[7]. The XCP software is also reported to use a lot of system resources regardless of an XCP CD being present in the CD-ROM or not, as well as leading to system crashes. The installation of the software is not mentioned in the EULA (End User Liscence Agreement) or when you insert the CD. Installing software without the user's knowledge is a bad practice, and ethically wrong. There has also been some claims that XCP is in violation of the GNU GPL liscence, as it links statically to code from the LAME MP3 library. The developers of LAME has, in an open letter to Sony/BMG [mai], stated that they will not sue, as long as Sony/BMG takes "appropriate action".

3.4 FairPlay

FairPlay is the DRM scheme Apple uses in their products, namely iTunes, iTunes Music Store and the iPod.

FairPlay uses the Rijndael algorithm to encrypt an AAC[8] stream, which it then embeds in an MP4[9] container file [Ano]. The key used to decrypt this stream is called the "master key", and is embedded in the MP4 file, but in an encrypted form. The master key is encrypted with what is called the "user key". The way this is used is the following:

1. A customer buys a song on the iTunes Music Store.
2. A master key is generated, and used to encrypt the AAC stream.

[5] Uninstall XCP

[6] PowerToys (with TweakUI)

[7] Sony/BMG settlement

[8] Advanced Audio Coding

[9] MPEG-4 Layer 14

3. A user key is generated, and used to encrypt the master key.

4. The encrypted master key and the encrypted AAC stream is embedded in an MP4 file.

5. The user key is stored in the iTunes Music Store.

6. The MP4 file is sent to the user.

The user key is retrieved by iTunes (the application), and stored in encrypted form on the harddrive, in a file called `SC Info.sidb` [Hal]. The encryption algorithm applied to this file is also based on the Rijndael algorithm: First, iTunes XORs the plaintext with the output from a proprietary pseudorandom number generator which is seeded by a system-dependent seed. This is then encrypted using the Rijndael algorithm, and a system dependent key. When the customer plays a song controlled by FairPlay, iTunes will look up the user key in it's key database. If it's not there, it contacts the Music Store to get it from there.

iTunes Music Store also keeps track of how many PCs a customer is using. If you want to play your purchased songs on another computer, the iTunes application on that computer will contact iTunes Music Store (where you authorize yourself using your "Apple ID" and a password), and request the user keys. For every machine the customer uses, a unique machine identifier is used when contacting the iTunes Music Store. An account can only be connected to five computers, so to be able to play your content on new machines, you will have to "deauthorize" old computers you are no longer using.

Several applications exists that enables users to strip the DRM from their purchased songs. Most of these use the same key iTunes use, and some of them even gets the keys from the server. The most popular program seems to be Hymn[10] (Hear Your Music Anywhere).

[10] Hymn Project

3.5 PlaysForSure / Windows Media DRM

PlaysForSure is a certification from Microsoft that is given to devices that support Windows Media Audio, the MTP[11] or USB Mass Storage, and the Janus DRM (or Windows Media DRM for Portable Devices), which is Microsoft's DRM scheme for portable devices. PlaysForSure is supposed to give the customer a guarantee that the device will "play nice" with other devices that has the same certification. It is also a guarantee that the device will be able to play content purchased in a PlaysForSure web store, such as Napster or Yahoo Music and MTV URGE.

Windows Media DRM, or WMDRM, uses a combination of elliptic curve cryptography key exchange, the DES block cipher, a custom cipher called "MultiSwap" (only used for MACs), the RC4 stream cipher and the SHA-1 hashing function. WMDRM is designed under the assumption that it will get cracked, and is therefore easy to renew through updates from Microsoft. It has, in fact, been cracked several times, but Microsoft has usually patched the "holes" after some time. All attempts to crack the DRM scheme seems to take the same route; instead of breaking the actual encryption, which will be to hard, they hook into the "black box" component to get the keys used for decrypting the files from the memory used by Windows Media Player.

In the case of music files, the files you are getting from a PlaysForSure music store, is ASF[12] container files. These files contains encrypted WMA[13] files, along with a "KID". A "simple" breakdown of how a file is decrypted follows [Vio]:

1. The KID is found in the ASF file.

2. The Media Player then looks up the KID in a local database, called `drmstore.hds`. If the KID is not found in the local database;

   (a) Media Player will load an URL found in the ASF file in an embedded IE[14] object.

[11] Media Transfer Protocol
[12] Advanced Systems Format
[13] Windows Media Audio
[14] Internet Explorer

```
<ENABLINGBITS>
  <ALGORITHM type="MSDRM"></ALGORITHM>
  <PUBKEY type="machine">
     S3O5h0DH*NdqFlK6W6InM2*5VxnnYtCCOuCwvOsXTdOCgZjtseE5iQ==
  </PUBKEY>
  <VALUE>
     VEsbPedfwrybrpkg0fhoOfe5eB9ef0R7QTxgX7NbtMIFK!h*4Pk7ek
     PUqlDIRqYwQkgCGEOr0qtQdCUYszT!b7XedCIpsApQjstaFmafahM=
  </VALUE>
  <SIGNATURE>
     KpxCm6lSXH8dTPI359jToftSEuLiP9v*zpHAy!kDEhlYkw6mkfQzlg==
  </SIGNATURE>
</ENABLINGBITS>
```

Figure 1: Example of the `ENABLINGBITS` tag

(b) The webserver may prompt you for a username and password, check if you are in the correct country, and may take other measures.

(c) The webserver will then, assuming everything is OK, send the liscence, which is then stored in the local database.

3. An XML document is retrieved from the database, and parsed to find the `ENABLINGBITS` tag.
4. The value of the `VALUE` tag is then decrypted by a private key found in `indivbox.key`.
5. The decrypted data is then hashed through SHA-1 to make a "SID".
6. The SID is then used as keys for DES, RC4 and "MultiSwap" to finally decrypt the WMA file.

Obviously, if the device trying to decrypt the audio file for some reason is not able to connect to the Internet, it will not be able to retrieve the liscence, and the device will be unable to play the file. An example of the `ENABLINGBITS` tag can be seen in figure 1.

The values used are only example values, but should give you a clue about how the document looks. The `PUBKEY` is decoded as Base64 data (modified for the character set used by Microsoft), and contains the public key for the machine used to decrypt the file. More details about decrypting the file can be found here [Scr] (note that this document is not quite current, but a lot of the details remains the same in the newer versions of WMDRM).

WMDRM allows content providers to specify things like expiration dates, which, at least not yet, is not possible to do with Apple's FairPlay scheme. As with FairPlay, several applications enables users to strip the DRM from their purchased songs. Some even allows you to strip the DRM from content you're "renting". Microsoft has released several updates to WMDRM to defeat the purpose of these programs. The most popular program seems to be FairUse4WM.

3.6 High-definition Digital Content Protection (HDCP)

HDCP is a cryptographic system designed to protect audiovisual content over HDCP supported interfaces. The system was designed with the next generation of High-Definition video storage systems represented by HD-DVD and the Blu-Ray discs. HDCP protects the last link in the playback process namely the connection between the playback device and the display device. Interfaces supporting the HDCP system are [LLC06] HDMI [15], DVI [16], UDI[17], GVIF[18] and DisplayPort.

3.6.1 HDCP authentication

The keys in the HDCP protocol are computed based on a pre-assigned secret sequence of numbers (Secret Device Key: SDK), and a Key Selection Vector (KSV). The two parties exchange their KSVs which tells the other party how to compute the secret key based on the pre-assigned SDK. Due to the mathematical properties of the KSV and the SDK two authentic parties will always get the same result from this process.

The first part of this HDCP handshake is illustrated in figure 2.

In figure 2 the transmitter initialises the communication by sending it's KSV (Aksv) and a psaudo-random salt number An. The receiver responds with it's KSV (Bksv) and the REPEATER bit which tells if the receiver is a device capable of retransmitting the data.

[15] High Definition Multimedia Interface
[16] Digital Visual Interface
[17] Unified Display Interface
[18] Giga-bit Video Interface

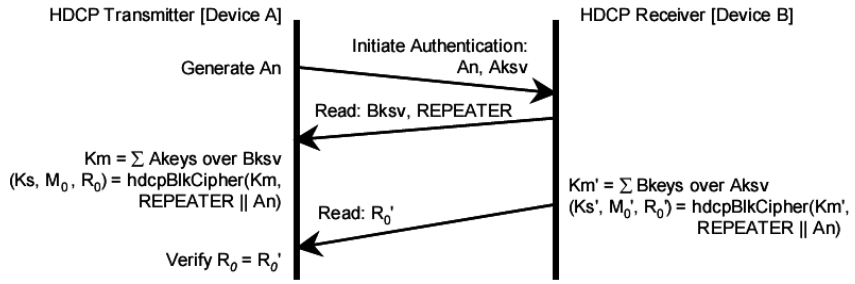

Figure 2: First part of the HDCP authentication process.[LLC06]

After receiving both calculate the secret key K_M and K'_M by adding the bits in the pre-known SDK by the rules given by the KSV. Then the transmitter calculates the values M_0, K_S and R_0, and the receiver calculates M'_0, K'_S and R'_0. These values are calculated using the HDCP block cipher `hdcpBlkCipher()` and K_M/K'_M as keys. The transmitter finally verifies that R_0 is equal to R'_0 in order to authenticate the receiver. Limitations are added to the protocol to complicate forgery with adding a time out on the computations done by the receiver. If K'_0 is not ready within 100ms the authentication will fail [LLC06].

If the receiving party sets the REPEATER bit, the process described in figure 3 must be completed before continuing. If the REPEATER bit is not set the authentication will just continue to the next part described in figure 4. After the repeater has authenticated itself, it collects all KSVs from the down-stream receivers (which also might be repeaters, doing the same procedure). This allows the transmitter to keep track of all down-stream devices. The repeater and the transmitter then respectively calculates the values V' and V, which the transmitter checks equality on, before accepting the list of KSVs. V and V' is calculated as a SHA-1 hash of the concatinated value of the values in the KSV list, the value M_0 from part 1 of the protocol, and the Bstatus value. Bstatus should be the value READY. Lastly the transmitter checks if any of the KSVs are on the key revocation list. Autentication failes if this is the case.

In figure 4 the last part of the authentication protocol is described. This part of the protocol is repeated over and over again during transmission. Both the transmitter and the receiver recalulates new cipher initialization values every 128^{th} frame. K_i, where i is the frame number, is the key used to initialize the cipher, both for encryption and decryption of each frame. This value is checked for equality at least once every 2 seconds, either asynchronously on a predetermined interval or syncronously every time the value changes (every 128^{th} frame). In order to better detect errors in the decryption the optional use of a the value P_j can be used. This value is the value found in pixel 0 in channel 0 of every 16^{th} frame XORed with the least significant byte of R_j, where j is the frame number. When the transmitter and the receiver both calculates this value, the receiver is able to check whether or not it's decryption is done correctly.

3.6.2 HDCP security

The biggest challenge for the developers is that neither of these devices can be trusted with knowing the keys. The obvious way of bypassing the security would be to reverse engineer a key from a device, and then use this key to simulate this device. That is why the developers of HDCP introduced a scheme called Key Revocation List (KRL). This is a list of Key Selection Vectors that is no longer accepted by the devices. This list was supposed to be used to block compromised KSVs, and hence delay illegal copying. As I will mention later this scheme will never fullfill it's original task.

The HDCP protocol is designed in such a way to maximise the number of colluding users needed to compromise the system. On the other hand, the developers of HDCP faced very limited storage resources. The resulting product was a compromise between the security requirements and the limited

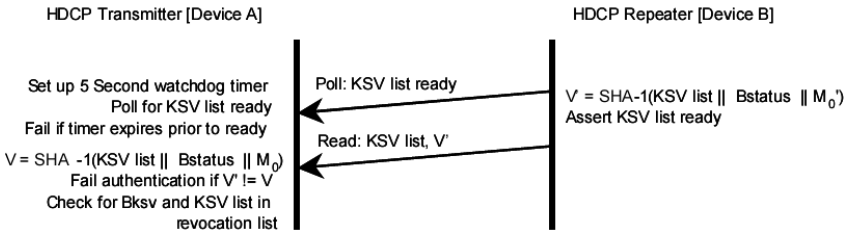

Figure 3: Second part of the HDCP authentication process. If the receiving party is a repeater.[LLC06]

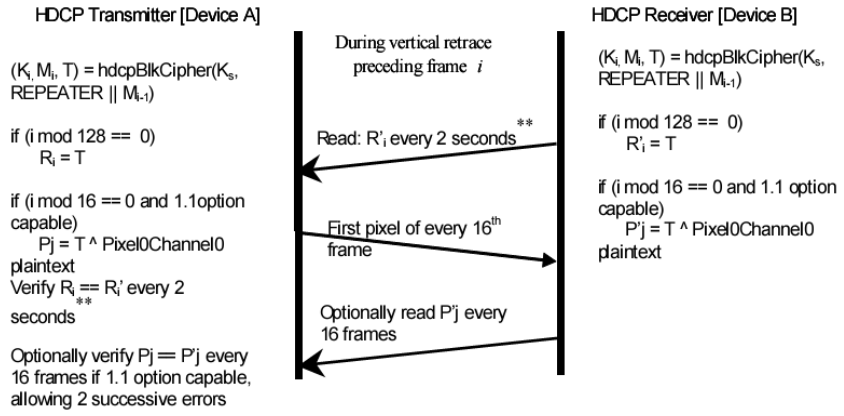

Figure 4: Third part of the HDCP authentication process.[LLC06]

storage capacity of the devices. The KSVs are based on a vector of only 40 numbers. The function of the vector is to decide which numbers from the SDK of each device are to be used in the key. The key calculation is done through a simple addition rule-set, which means that for those interested in breaking HDCP you wil only end up with a pretty straight forward linear equation with only 40 unknowns. Given knowledge of 40 devices' SDK it is not only elementary mathematics to crack one device's key, but it is possible to break every possible device.

The difficult part then, is of course, where to get the hold of the 40 SDKs. Obviously the designers of HDCP assumed that getting the hold of these SDKs would be a big problem for the crackers. The SDKs are protected through obscure hardware design, or obfuscated code where HDCP are implemented, but the possibillity of reverse engineering 40 SDKs from either hardware or software implementing HDCP on a global basis is quite large. Also considering the fact that the HDCP authorities sell packages of 10,000 SDKs for the neat sum of $ 16,000 it is not a suprise to those familiar with some cryptographics or even just basic mathematics that HDCP did not survive long [CGJ+].

4 Conclusion

[RC] lists several reasons why the DRM schemes of today are inefficient, amongst others several shortcomings of the mainstream operating systems that makes DRM impossible to implement in the way it is intended (ie. that the users are unable to circumvent it). It also states that the addition of Trusted Computing[19] does not help remedy the fact that DRM needs a "trusted path" to be "safe". This is functionality which must be implemented in all the programs the content passes through. Even when trying to create a dedicated system in hardware for protecting high-definition media, DRM failes miserabely. Of course one can speculate that the simplicity of HDCP was intended as a way to create a better, less complicated DRM, but in the attempt they also manage to remove any kind of noteworthy protection of the media.

DRM is, as argued by many in the technological world, mostly a pain in the butt for those buying and using media in a legitimate way, and in most cases no guarantee that those interested in illegitimate distribution or alterations of the media are even delayed in their work.

[19] Trusted Computing Group

References

[Ano] Anonymous. Hymn manual. URL.

[CGJ+] Scott Crosby, Ian Goldberg, Robert Johnson, Dawn Song, and David Wagner. A cryptanalysis of the high-bandwidth digital content protection system.

[Hal] J. Alex Halderman. Hidden feature in sony drm uses open source code to add apple drm. URL.

[Inf] Home Theater Info. Dvd regions. URL.

[LLC06] Digital Content Protection LLC. High-bandwith digital content protection system. Technical report, 2006.

[mai] LAME maintainers. Open letter to sony bmg (and its owners, sony and bertelsmann), first4internet, and the lame community. URL.

[RC] Jason F. Reid and William J. Caelli. Drm, trusted computing and operating system architecture. URL.

[Scr] "Beale Screamer". Microsoft's digital rights management scheme - technical details. URL.

[Ste] Frank A. Stevenson. Cryptanalysis of contents scrambling system. URL.

[Tou00] David S. Touretzky. Gallery of css descramblers, 2000. URL.

[Vio] Viodentia. Drm hacking question. URL.