A Key Management Architecture for Digital Cinema

Sebastian Knop[1], Jan Fröhlich[1], Roland Schmitz[2]

[1]CinePostProduction GmbH, Munich, Germany
{sebastian.knop@cinemedia.de, jan.froehlich@cinemedia.de}

[2]Hochschule der Medien, Stuttgart, Germany
schmitz@hdm-stuttgart.de

1 Introduction

The advent of digital cinema technology called for the creation of open standards to ensure high technical performance, reliability and interoperability of these systems. This call was answered by the Digital Cinema System Specification [1], which was first published in July 2005 by the Digital Cinema Initiatives LLC, a joint-venture of the motion picture studios Disney, Fox, Metro-Goldwyn-Mayer1, Paramount Pictures, Sony Pictures Entertainment, Universal Studios and Warner Bros. Studios. The specification regulates formats, interfaces, equipment behavior and testing procedures, affecting every stage from the creation of the final content distribution package at a production studio to the actual exhibition at a theater. It has since become the de-facto standard for the rapidly growing number of installed digital cinema systems.

Content protection is provided by a digital rights management (DRM) system described in the specification, making use of asymmetric key cryptography and digital certificates. The renting of analog film reels is emulated by content decryption keys with limited validity periods. Thus, the generation and management of those keys becomes part of the service that has to be offered by content providers, production studios or distributors.

Compared to the development in the USA or UK, the adoption of digital cinema technology by theaters in Germany has been slower. Nevertheless, the numbers are increasing fast. From the total of approximately 4800 screens in Germany [2], 164 screens were equipped with digital projection technology in June 2008 [3]. By December 2009, an approximate total of 350 screens had already been switched [4], and by October 2010 the estimated amount grew over 800 screens.

In this context, the post-production studio CinePostproduction GmbH wanted to expand its service portfolio to include the generation of DCI compliant keys. For the successful integration of such a facility into the production workflow and to attend to the business requirements, a key management architecture is required.

2 DCI Security Architecture

The security architecture described in the DCI specification pursues a number of high level goals:

- Enable decryption and playback of content based on a set of rules agreed upon by the involved business entities,
- protect content against unauthorized access, copying, editing and playback,
- keep records of security related events,
- define the security architecture in an open fashion and provide a set of standards allowing the implementation of compliant equipment by third parties.

The stated goals are achieved primarily through the use of content encryption and a digital rights management system focused on the exhibition environment. To enable the decryption and playback of content according to the defined rules in such a system, decryption keys and further data must be transported from content provider to the exhibitor in a secure way. For the transport of keys and the DRM related data, the specification defines the Key

Delivery Message (KDM) format (see section 2.4). Each KDM has a specific validity period, the time span during which a Security Manager (see section 2.2) will authorize the decryption and playback of the associated content

2.1 Protecting Track Files and Reels

The DCI packaging (DCP) format is based on a hierarchical system. The smallest units in this system are track files, carrying a single essence of actual content like image, audio or subtitle data, plus metadata. A compliant track file is a self-contained instance that can be interpreted by a compliant decoder.

A track file starts with a file header and ends with a file footer. The track file body consists of several KLV (key length value) units. The key is an identifier for the content type of the KLV unit, length specifies the length of the value. When encryption is used to protect the content, only the essence is encrypted, but the original KLV triplet is embedded into a new KLV with adapted metadata, additional encryption data and the message integrity code (MIC), which allows every frame to be checked for consistency.

The DCI specification defines JPEG2000 [5] as an intra-frame codec. Although there is a standard called JPSEC [6] covering JPEG2000 compliant encryption, the DCI specification uses a different approach: Encryption is performed on the essence KLV packets covering one frame (or data with the duration of a frame) each, using the AES block cipher algorithm in cipher block chaining (CBC) mode with a 128 bit key, as defined by NIST in [7].

At the next level in the package hierarchy are reels. The reel concept originates from the physical reels of analogue film typically covering 10 to 20 minutes of play time, but it was carried over into the digital workflow as a means of segmenting a feature film into multiple parts for easier handling during post production.

A reel is required to consist of at least one track file. Playback devices are required to be able to play back content segmented into reels without artefacts and without interruptions on reel boundaries. When encryption is used, one key per reel and per track file shall be used.

2.2 Playback Environment

In the playback environment, the functional entity that processes the keys and that is trusted to enforce the rules of the DRM system is the Security Manager (SM). It is a logically separate component of the architecture, although it is intended to be implemented as part of the playback server. SMs and other security critical components, called security entities (SE), are placed inside a Secure Processing Block, which provides physical security to the relevant hardware.

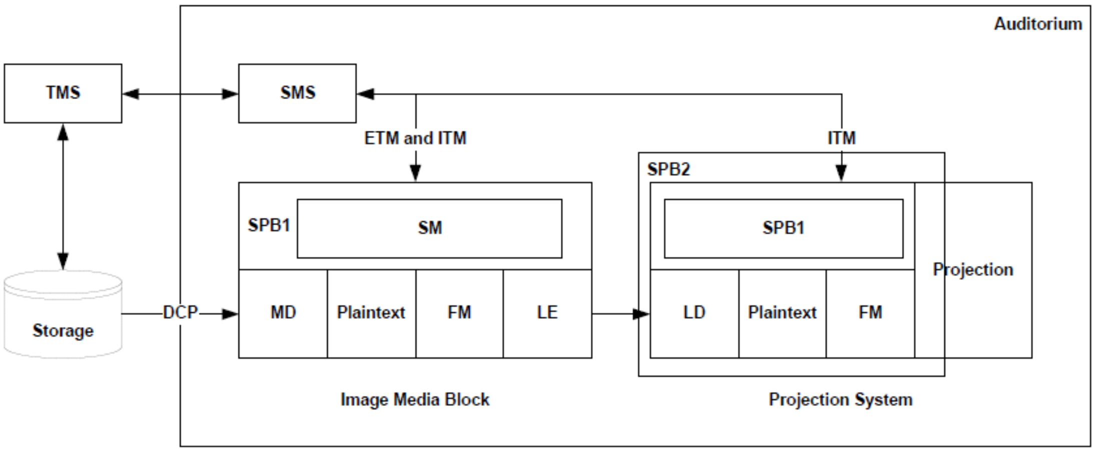

Figure 1: Digital Cinema auditorium security components

Figure 1 shows a schematic overview of the security components of a theater auditorium. The basic functions of the shown components are as follows:

- The Image Media Block (IMB) is a Secure Processing Block (SPB) which shall contain a Security Manager, Media Decryptor (MD), Forensic Marker (FM), and a Link Encryptor (LE) module.
- The Projection System is a Secure Processing Block (SPB) containing a Link Decryptor (LD), an optional Forensic Marker and the actual optical image generation system. Image Media Block and Projection System may be implemented in an integrated fashion (not shown in the figure); in that case the LE and LD are omitted.
- SPB1 and SPB2 denote two physical protection levels for Secure Processing Blocks, where SPB1 offers a higher level of physical security than SPB2.
- The Security Manager (SM) is the security entity trusted to enforce the rules of the DRM system and provides the decryption of the content decryption keys as well as authentication and control over secure processing blocks.
- The Media Decryptor (MD) is the component responsible for decrypting the encrypted content with the content decryption keys.
- The Forensic Marker (FM) is responsible for embedding a digital watermark into both image and video. If the Image Media Block contains a FM, the FM in the Projection System is optional.
- The Link Encryptor (LE) and the Link Decryptor (LD) are the modules responsible for encrypting and decrypting the content for transport outside of SPBs over a wired connection.
- The Extra Theater Message (ETM) and the Intra Theater Message (ITM) are generic formats for security related messages.

2.2.1 The Secure Processing Block (SPB)

The SPB is a generic container with a defined perimeter providing physical protection for security critical system components such as trusted entities like the Security Manager or modules handling content in plaintext form. This is intended to offer protection to secrets not secured by cryptographic mechanisms.

SPBs shall carry an RSA private key [8] and a matching digital certificate stating their role, as to allow authentication by a Security Manager or another SPB. After physical installation, equipment with interacting SPBs needs to undergo a process called “marrying”, in which the SPBs mutually exchange and store their certificates after a human supervisor confirmed a correct interconnection of the systems, which is a prerequisite for a secure TLS connection with mutual authentication between SPBs.

According to the DCI specification, SPBs shall have the following characteristics:

- Tamper evident: Breaches of the security perimeter shall permanently alter the equipment, which should be noticeable upon inspection.
- Tamper resistant: Breaching the security perimeter to expose the contained secret shall be difficult and require considerable effort and tools.
- Tamper detecting and responsive: The security perimeter is actively monitored, triggering the erasure of the protected secrets in case of a breach.

2.2.2 Link Encryption

Link encryption shall be used in all cases where plaintext content is exchanged between SPBs. In this case, the Image Media Block (IMB) shall provide the Link Encryption keys to be used, to both the Link Encryptor and Link Decryptor. The specification requires the use of either Triple DES with 112 bit keys [9], or AES with 128 bit keys. For each transfer of encrypted content, a new set of encryption keys is mandatory.

2.2.3 Forensic Marking

The specification requires that each IMB shall be equipped with a forensic marking module. Watermarking does not prevent unauthorized copying or playback of content. However, it allows for tracing leaked content back to the installation which performed the original playback, helping in taking measures to prevent future leaks [10].

The DCI does, however, not define the use of a specific watermarking technology, rather formulating a list of requirements and leaving the implementation up to the equipment manufacturers. The list of requirements includes the following points:

- The watermarking technology shall allow the embedding of at least 35 bits of data. Of these, 16 bits are used for a timestamp with 15 minute granularity, covering the span of one year before repeating. The remaining 19 bits are used to code a serial number, which shall uniquely identify the FM module and thus the playback system used. The 35 bits of data shall be embedded in every 5 minute segment of the playback.
- Each Forensic Marker instance shall be assigned a unique serial number (FMID), which cannot be reprogrammed without breaching the security perimeter provided by the type 1 SPBs.
- Providers of watermarking technology shall offer it under reasonable and non-discriminatory (RAND) terms.
- Detection shall be possible within the premises of the rights owner.
- The watermark shall be robust enough to survive a number of signal processing attacks, image/audio transformations, format conversions and recording by consumer electronics.

There is a number of suitable video watermarking methods available (see e.g. [11]).

2.3 X.509 Digital Certificates

Digital certificates are a cornerstone of the DCI security architecture. They enable a series of security functions like encryption, digital signatures, authentication and the implicit transfer of trust. At the core of the certificate concept is the RSA key pair. This pair of keys enables the elementary asymmetric key cryptography functions, specifically encryption with the public key, decryption with the private key and signing with the private key, verifying with the public key.

The DCI architecture makes use of digital certificates in the X.509 version 3 format [12], with a clearly defined set of characteristics. For example, it is specified that the RSA public key modulus shall have a length of 2048 bits. The employed signature algorithm shall be SHA-256 [13] with RSA key encryption.

The concept of using trusted third parties to sign certificates enabling the transfer of trust can be extended to a scenario, where whole chains of certificate signing are used. They form a tree-like structure, where all certificates are directly or indirectly derived from one single certificate, the root of trust. In this context, the entity controlling the signing certificate is called Certificate Authority (CA). The certificate at the top of the signing tree has to be self-signed and is called the Root CA certificate. Certificates that sign other certificates, but are not self-signed, are called Intermediate CA certificates. Finally, the certificates at the edge of the tree, that do not sign any other certificates, are called Leaf certificates.

If implemented consistently, the number of certificates, that other entities have to know as trusted in order to be able to verify other incoming certificates, can be reduced to one, or few, depending on how many certificate trees are being used. The signature of a received certificate is verified, and then the signature of the signing certificate is verified. This is repeated recursively until the root certificate is reached, confirming the authenticity of the whole chain.

The specification requires that the employed certificate chains have a length of at least two beyond the leaf certificate, meaning at least one intermediate certificate between root and leaf. For the verification of certificate chains, the specification cites numerous properties to be checked. For instance, the verification of the validity shall be performed in the chain context, where the validity of the signed certificate must fit completely into the period of the signing certificate. Signing is only allowed by certificates that carry the appropriate flags and role specifications.

2.4 Key Delivery Message (KDM)

The KDM is a specialized instance of an ETM used to transport content decryption keys and DRM information to a SM. KDMs are always targeted at a specific recipient. This is achieved by means of encryption using the public key of the recipient's certificate. This way, the content decryption keys are protected and only the holder of the matching private key is able to extract them from the KDM.

3 The Key Management Architecture Project

3.1 Requirements

The existing workflow for KDM ordering at CinePostproduction GmbH, henceforth referred to as CPP, can be described as follows: When an employee of a movie theater orders a DCP and KDMs for his theater server(s) from his distributor, the distributor in turn sends an order to CPP's distributor scheduling office by fax or email, where it is entered into an ERP system. The order is accompanied by a digital certificate file from the theater server(s) for which the KDM(s) should be generated. For the creation of DCPs, CPP employs a mastering system being able to encode and package audiovisual content into distribution-ready DCPs. Additionally, a limited number of KDMs, also known as “distribution KDMs” or “master KDMs”, are generated, to be used for deriving the KDMs for the actual playback on the theater servers. In the past, however, CPP lacked a suitable system for KDM creation and had to rely on a third party to generate them. Therefore, a project was initiated with the following requirements:

- A system for the generation of DCI compliant keys (KDMs) shall be implemented, along with an architecture for the management of keys and digital certificates, which is to be integrated into the existing workflow of the company.
- A security concept to protect the sensitive data contained in the systems shall be designed, while minimizing its operational impact on the usability.
- The implementation should build on open source software and other freely available core technologies.

3.2 Design

The key management architecture project was christened CinDi, as an acronym composed of the initials of “cine”, “digital” and “distribution”. The design followed the idea of using an existing web platform developed in-house, which is currently used to deliver movie trailer DCPs to theaters, as well as for providing video samples of post production work to customers, as a platform for the management of KDMs. For security reasons, the process of generating the KDMs should be separated from it. This way, the web platform could be extended to provide the user interface and all the necessary key and order management features, while the actual KDM generation and security critical processes are performed with a separate software component running on another, protected system.

For the communication between systems, an interface based on HTTP and XML was designed. The system hosting the KDM generator is protected by an additional firewall. This firewall between the KDM generator and the web platform is configured to allow only allow traffic between them and only outgoing HTTPS connections from CinDi, blocking all other forms of traffic and traffic origins.

Once implemented and integrated into CPPs workflow, CinDi will change the existing ordering process, as described in section 3.1, to a new process (see Figure 2) differing from the existing one the following points:

- When DCPs are created, the CPLs and master KDMs generated along with them are imported in the web platform.

- After the distributor scheduling office receives an order for a KDM, the relevant order details are entered in the web platform, together with the target certificate.
- The KDM generator picks up the order package, generates the KDM(s), and posts them back to the web platform in a delivery package.
- The web platform sends the KDMs to the specified recipients via email.

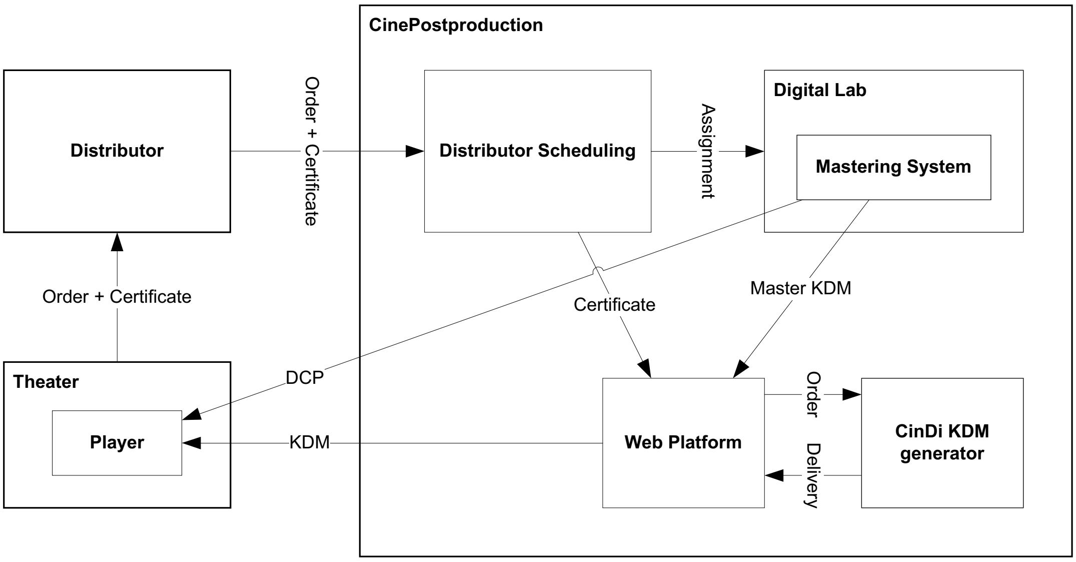

Figure 2: The Workflow for Generating KDMs

3.2.1 The KDM Generator

The KDM generator is the key component of the CinDi architecture. It takes a master KDM, a target player certificate and an order description as input and generates a KDM for the referred target player as output.

The architecture was designed in such a way that the existing web platform handles all data persistency and user interactions. The KDM generator does not store output except for log entries; all relevant output files and messages are sent back to the web platform. Except for the initial configuration, the only duty that needs to be performed by a user on the KDM generator is managing the set of trusted CA certificate chain files. The intent of this design is to allow the KDM generator to run mostly autarkic and isolated for security reasons, requiring as little as possible user intervention. The KDM generator contains private keys that enable the decryption of the content decryption keys in the master KDM, and should therefore be protected from access. By reducing the complexity of the KDM generator software, the potential for security flaws is reduced.

On the Linux based host system, the Cron scheduler daemon or comparable mechanism is configured to initiate a script of the KDM generator in regular intervals. Once activated, the script polls the web platform for new order packages. If new orders are available, an order description is created and returned to the KDM generator.

If the manual mode of operation of the KDM generator is being used, order packages may be processed, being supplied over the local web interface or copied to the local file system and given as parameter on the execution via CLI.

The received order package is processed by the KDM generator. This includes the validation and signature verification of the order description. The target player certificate sent with each individual KDM order is validated against the procedures of the CTP, and then verified against the set of trusted certificate chains. Having passed those tests, the order is dispatched to the KDM engine, which generates a new KDM based on the provided master KDM, the order description and the target player certificate.

Once all orders have been processed, a delivery package with description and signature files are generated posted back to CPPs web platform.

3.2.2 Realization

The main programming task in the CinDi project was the implementation of the KDM generator. The very first steps taken in the implementation were some experiments done to determine if PHP would be adequate as a programming language. Python and Java were also briefly considered, but PHP was the first choice for a number of reasons: The web platform is already written in PHP and since it will interface with the KDM generator, some code reuse was expected to be possible.

The most critical feature set that had to be investigated were the available cryptographic functions and the capabilities for handling X.509 digital certificates. Upon investigation it was determined, that the Hash and the OpenSSL extensions combined would cover all necessary elementary cryptographic functions like the creation of hashes (SHA-1 and SHA-256), encryption and decryption with both symmetric (AES 128 bit) and asymmetric (RSA 2048 bit) keys, and the generation of pseudo-random numbers. Improvements and additions to these functions made PHP version 5.2.4 the minimum requirement for CinDi.

The handling of digital certificates is also enabled by the OpenSSL [14] extension, but it proved to be incomplete compared to the feature set of the command line version of OpenSSL or even the functions exposed in the API of the OpenSSL cryptographic library. For instance, when reading certificates, not all necessary information can be obtained with the PHP extension. When creating certificates, the extension only offers rudimentary controls over the characteristics of the created certificate. Another required but unavailable function is the extraction of specific sub strings of the binary representation of the digital certificate, including the segment holding the public key, or the to-be-signed portion of the certificate. It was determined, however, that it would be possible to implement unavailable functionality. With this assessment the choice for PHP as programming language for CinDi was confirmed.

4 Open Issues in the DCI architecture

Originally the DCI consortium intended to implement a central CA for certificate signing and tie the distribution of certificates to third parties to successful standard compliance certification of the target devices. That way all certificates could easily be verified against a single trusted CA certificate and it would insure that only compliant devices would be allowed and thus utilized under the DCI label.

However, because of monopoly concerns, political disagreement between the members of the consortium and in favor of adoption rate, this intended design was never put into practice, leaving the specification in a state where multiple CAs are allowed and where any CA is trusted by default.

This has led to a number of issues. Most equipment manufacturers now act as their own CA, and since certification became optional, the requirements for standard compliance became less strict. In fact, equipment manufacturers defined an informal interim standard called “Interop” with a feature subset of the DCI and image storage based on MPEG2 instead of JPEG2000, to be used while DCI compliant equipment is being developed. As of this writing, only some projectors have been certified DCI compliant [15], while the first compliant Media Block is expected to be certified this year. “DCI” is not a registered trademark, so “DCI capable” equipment has been sold on the market before regardless.

For content mastering studios or entities creating KDMs this means added complexity for the verification of target theater certificates, which is essential to prevent the supplying of content decryption keys to entities with a faked identity. Since no central CA exists, studios have to maintain their own pool of trusted CAs and implicitly trust the equipment manufacturers and their equipment without being able to rely on a certification asserting their security.

Since CAs are trusted by default, a hypothetical attacker could forge a player certificate and certificate chain and send them both to a KDM creating entity (like is common practice during legitimate KDM ordering). Without proper trust verification against the set of known and trusted CA certificates it might wrongly create the requested KDM, thus compromising the content in the underlying DCP. It also enables another scenario, where a rogue content supplier distributes KDMs and/or DCPs after having gained access to the content by other means, since theater servers wouldn’t reject those.

While the specification is now being standardized by standardization bodies like ISO, unfortunately the standard *development* efforts have mostly stopped, only adding special case support like stereoscopic images and other frame rates for archive footage. This essentially leaves the security of the DCI architecture in an inconsistent state, where content suppliers are left to deal with the resulting issues.

5 Conclusion

In the present contribution, we have given an overview of the DCI security specification and we have described an approach to integrate the generation of DCI compliant Key Delivery Messages (KDMs) into the workflow at CinePostProduction GmbH. Some open issues concerning management of certificates, especially the default trust placed in CA certificates, were discussed, leading to the possibility of certificate chain forging and a rogue content supplier scenario.

Since it is expected that the DCI specification will not be updated to rectify the issues with lacking mandatory certification, central CA and improper certificate trust verification, content providers have to develop methods for ensuring content security. An approach that is being considered to achieve that is a trusted device list (as certificate white and black list), either held by a neutral entity or shared among studios.

References

[1] Digital Cinema Initiatives, LLC. Digital Cinema System Specification, Version 1.2. Digital Cinema Initiatives. http://www.dcimovies.com/DCIDigitalCinemaSystemSpecv1_2.pdf.

[2] Deutsche Filmförderungsanstalt. Marktdaten - Kinosaalbestand 2005 bis 2009. http://www.ffa.de/start/download.php?file=marktdaten/1_Fuenf_Jahre_Blick/04bis09_jahresabschluss.pdf.

[3] European Audiovisual Observatory. Focus 2009 - World Film Market Trends. European Audiovisual Observatory. http://www.obs.coe.int/online_publication/reports/focus2009.pdf.

[4] CineMedia Film AG. Geschäftsbericht 2009. http://www.cinemedia.de/download/cinemedia_geschaeftsbericht_2009.pdf.

[5] ISO/IEC. ISO/IEC 15444-1 Information technology - JPEG 2000 image coding system. 2004.

[6] ISO/IEC 15444-8. Information technology—JPEG2000 imagecoding system, Part 8: Secure JPEG2000. 2007

[7] National Institute of Standards and Technology (NIST), Special Publication 800-38A. 2001.

[8] R. Rivest, A. Shamir, L. Adleman: A method for obtaining digital signatures and public-key cryptosystems, Communications of the ACM, Vol. 21(2), p. 120-126, 1978

[9] National Institute of Standards and Technology (NIST), Federal Information Processing Standards (FIPS) Publication 56-3, http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf

[10] I. J. Cox, M. L. Miller, J. A. Bloom: Digital Watermarking, Morgan Kaufman Publishers, 2001

[11] J. Lubin, J.A. Bloom, H. Cheng: Robust, Content Dependent, High Fidelity Watermark for Tracking in Digital Cinema, in: P. W. Wong, E. J. Delp (Eds.): Security and Watermarking of Multimedia Contents V, Proc. SPIE Vol. 5020, 2003

[12] D. Cooper et al: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, IETF Proposed Internet Standard, http://tools.ietf.org/html/rfc5280

[13] National Institute of Standards and Technology (NIST), Federal Information Processing Standards (FIPS) Publication 180-2, http://csrc.nist.gov/publications/fips/fips180-2/fips180-2withchangenotice.pdf, 2002

[14] http://www.openssl.org/

[15] Digital Cinema Initiatives, LLC. Compliance Test Plan – Compliant Equipment http://dcimovies.com/compliance/