MIECT: Security 2015-16

IEDCS: Identity Enabled Distribution Control System

Presentation: October 20th and 21st
Deadline: Nov 13rd (1st sub-project), Dec 20th (final project)

Changelog

- v1.0 - Initial Version.

Introduction

The distribution of several files, such as books, videos or musics is frequently conditioned by their authors. The purpose is to limit access to the content based on a particular distribution or reproduction license. Selling licenses tailored to each client consists on the monetization strategy, generating money to pay the creation and distribution of the content, as well as to extract profit from related commercial activities. As an example, a film may be sold to reproduction in a single country, or in specific formats such as DVD or BluRay. In more restrictive cases, such as the ones frequently used to distribute music file or eBooks, a given file is associated to a specific individual and a limited number of devices the individual owns.

The methods that enforce the reproduction and distribution restrictions is named a Digital Rights Management (DRM) system. There are many of such systems available, integrated in online services such as Apple Music, Amazon Kindle, or in classic distribution media such as BluRay and DVD. In DRM, it is assumed that the distribution channel or reproduction equipment tends to be non-cooperating. Therefore, policies are enforced not through simple decisions ('yes you can play the file') based on data available in a database, but through extensive use of cryptography to cipher and assess the correctness of content, and devices and software components.

In most cases, the system is composed by a set of servers that make content available, authenticate users, devices, authorize reproduction, and finally distribute the keys that allow proper deciphering of the content (or parts of the content). The device or software component that decodes the

file must be aware of the process to authenticate the user, obtain the correct keys and decipher the content. It is common for these components to be certified by the content distributor, which then attributes additional keys that may be specific for each device or software component.

One widely used example is the one used in the distribution of DVDs and BluRays. In this case, the distribution channel is secured using commercial agreements between a chain of companies, from the author to the retailer. Inside the BluRay media, content is marked with region code indicating where it can decoded (3 zones plus FREE), and is ciphered with a key title key. In order to obtain the title key a player must use its shared key (pre-distributed), a sequence key, and a volume specific key[1].

DRM processes are never simple and there are many opposing movements due to the restrictions it imposes or the problems they create (player update, unavailability of DRM servers, lack of freedom over owned content). Still, DRM is vital for the wide distribution of Intellectual Property restricted material. Moreover, it presents a great example to exercise the application of authentication, authorization, key distribution, and cipher methods, making this concept the objective of this project.

Project Description

The goal of this project is to develop an end-to-end (E2E) Digital Rights Management (DRM) system, named Identity Enabled Distribution Control System (IEDCS), for sharing protected files such as media or documents. In this specific case we will consider that the player can communicate with the DRM servers. As it is typical in a DRM system, media objects are not publicly available and must be acquired (e.g., at a Music Store), users are then allowed to play the media files with some restrictions: only in X (2) devices, a limited number of times, at a given time of the day, or in a determined region. Any other policy can be designed and implemented according to licensing and distribution restrictions. Students are free to choose what content to distribute and what the IEDCS will protect. eBooks in plain text format is perfectly acceptable[2]. An eBook player will essentially scroll through plain text files.

0.1 System Components

Implementation of this project will consider the existence two components: the IEDCS Server and the IEDCS Player.

[1] http://www.aacsla.com/specifications/

[2] Any text file is acceptable. For real ebooks in text format, please check http://www.gutenberg.org

The first component of the system is the IEDCS Server, which is the main key and content distribution point. It will provide a simple store frontend to users through which they can purchase a title, and an API through which the IEDCS player will interact. The store frontend can be implemented in any language, providing a web interface or making use of a set of command line tools. The objective of this project is the development of the correct cryptographic mechanisms and not of carefully designed user interfaces. The API should be provided over a plain socket, an HTTP REST endpoint, or any other similar method allowing the exchange of messages with the IEDCS Players.

The IEDCS Server should provide secure APIs to players so that no text is ever transmitted in clear. This means that the IEDCS server must use encryption to make their messages private and authenticated. If using a web interface, the deployment of a PKI using X509 certificates is perfectly accepted and recommended. Students should make sure that there is strong mutual authentication between Client and Server, and the messages exchanged, using asymmetric cryptography.

Users are registered with the IEDCS Server, having some cryptographic material stored. In particular, a unique User Key will allow ciphering content specifically for that user. Information from the PTEID Smart Card may also be stored if required.

The second component is the IEDCS Player, which has the main task of reproducing a particular file. For eBooks, the player will decode files accordingly and show the content to the screen, one page at a time. Development of DRM enabled players is frequently not a free activity, and this will also be the case of the IEDCS system. Player implementations must be validated by the IEDCS distributors. The IEDCS server knowns all implementations produced and can determine the integrity of their code.

Each player implementation will have a unique identifier, and some cryptographic material, in the form of a shared key (the Player Key). Also, when executing for the first time it will derive a unique identifier that is in some way tied to the underlying hardware (e.g., Ethernet MAC addresses, devices present, CPU, serial numbers of the mother board, etc...), and a Device Key which it stores in a secure manner. This key is sent to the IEDCS Server together with the device identifier, and other relevant data. Sending a user friendly name can also be helpful (e.g., Peter's Laptop). The secure storage of the device key can be dependent of a user password[3].

One of the requirements of the player is that it stores no cryptographic material besides the one stated before (Device Key, Player Key). If the player uses a key for decoding a file, the player should erase that key from memory after it was used.

[3]Using a function for secure derivation of keys from passwords is recommended

0.2 Buying and validating the title

The IEDCS Server will provide titles, which may have one or more files associated (e.g., Volume 1, Volume 2). In its initial state, a title is only a set of clear text files with some name, author, production date and identifier. Users may list all titles available and then acquire one file in particular. Each transaction should create a uniquely ciphered file, and a set of auxiliary tokens (or headers) containing identification, authentication and cryptographic information. When a title is bought, it should be ciphered with a unique key (the File Key) to secure the title content, and the correct set of auxiliary tokens is created.

The authentication token contain information that is used to verify the identity of the User and of the Player reproducing a specific file. For validating the Player, the IEDCS Server make check its integrity and authenticity through the cryptographic material that was initially hardcoded. The PTEID Smart Card is the medium used for User authentication, and the process should be mediated by the IEDCS Player. For this purpose, students should select which information to include in this auxiliary data in order to authenticate both parties. If no user information is present, no User authentication is required. This token exists only at the IEDCS Server.

The authorization token should specify which information the IEDCS Server will verify in order to restrict the reproduction of the file. The content of this token varies according to the file policy, and effectively instructs the Player to collect and send data specific data to the IEDCS server. Students should create a policy that is aware of the location, operating system, player identification, player integrity, system identification, and time of the day. This token exists only at the IEDCS Server.

The identification header provides the information essential for the Player handle the file, and to show its information to the user. This header should be sent together with the file.

Finally, the cryptographic headers contains the cryptographic information that allows deriving the correct File Key and decode the file. This header should be sent together with the file. Students should devise the appropriate cryptographic structures so that the File Key is only obtained after a chain of cryptographic operations with the following properties:

- No key is ever transmitted, either ciphered or in clear text. The File Key must be computed by the Player, which reduces the risks of a MiTM[4] attack.
- There should be at least one exchange between Player and Server. Implying that the Player cannot decipher the file without proper cooperation from the EIEDC Server.

[4] `https://en.wikipedia.org/wiki/Man-in-the-middle_attack`

- The Device Key and Player Key (exist at both components) are used to derive the File Key. Effectively associating the file with that combination of Player and Device.
- The User Key (exists at the IEDCS Server) is used to derived the File Key. Effectively associating that representation of the file with a specific user.
- Deploying the file to another device or player can reuse the same File Key. Implying that only the headers need to be changed.
- The correctness of the flow is only dependent of the correctness of the cryptographic operations. Implying that the IEDCS Server makes no decision based on the fact that it has sold a file to a user (it doesn't decide to accept the player based on a simple query to the database).
- Capture of a single interaction cannot allow reproduction of the file. Implying again that values are computed.

Please take in consideration that the IEDCS server has access to all cryptographic material. In this sense, the File Key can be the result of a computation, and not truly random (or it can be random). As a tip, remember the principles behind challenge based authentication (CHAP) and of 3DES (DES-EDE).

All data exchanges should be explicitly ciphered and authenticated using asymmetric cryptography. Hybrid cipher mechanisms can be useful to improve system performance.

0.3 Additional Security Measures

It is of vital importance that the IEDCS Server is secure against common attacks. Therefore, students must devise the methods to secure the execution of this component. In particular, the following guidelines should be followed, and the students should present how they address the issue:

- The IEDCS Server must not be vulnerable to SQL Injections, Buffer Overflows and XSS Attacks.
- All media files stored in the server should never be stored in the hard disk without encryption.
- Compromising the IEDCS Server should never result in system wide compromise.

0.3.1 Project phases

The project implementation should be split into two phases, corresponding to the two deliveries predicted. In the first phase, the students should consider all aspects related to distribution and reproduction of the media files, including the decryption of the files and key distribution aspects.

The second phase should add the support for authorization and authentication through the Portuguese Identification Card, enforcement of the reproduction authorization policies with authenticated policy information, and add the support for the additional security measures.

Delivery Instructions

You should deliver all code produced and a report before the deadline. That is, 23.59 of the delivery date. The delivery dates are November 13th, 2015 and December 20th, 2015.

In order to deliver the project you should create a project in the CodeUA[5] platform. The project should be named after the course name (`security2015`), the practical class name (e.g. `p2`) and the group number in the practical class (e.g. `g5`), with the following complete format for the examples given before: `security2015-p2g5`). Please commit to this format to simplify the evaluation of the projects! Each project should be given access to all the course professors.

Each CodeUA project should have a `git` or `svn` repository, and folders for each milestone (m1, m2,...). The repository can be used for members of the same group to synchronize work. After the deadline, and unless otherwise requested by students, the content of the repository will be considered for grading.

The report should address **all the studies performed, all the decisions taken, all the functionalities implemented and all known problems and deficiencies**. Grading will be focused in all these topics, and not only on the code produced!

Using materials, code snippets, or any other content from external sources without proper reference (e.g. Wikipedia, colleagues, StackOverflow), will imply that the submission will not be considered for grading. If referenced, external code will not be considered a contribution made by the students (will not be graded).

1 Grading

Grading will take in consideration the capabilities of the software delivered, as well as the elegance of both the design and actual implementation.

[5] `http://code.ua.pt`

2 Bonus Points

Up to 2 (two) Bonus points will be awarded if the solution implemented correctly supports extra features of interest, conceived by the students. If you wish to apply for bonus points, discuss this with the course professor.