# Project 1 - DVD Manufacturer

## Introduction to Computer Security

**Floriane Magera** (S111295)
**Fabrice Servais** (S111093)

March 27, 2015

# 1 Introduction

# 2 PlayerKeys

This section focuses on the keyfile generation. First we will explain how we derived the keys, and then what system we used to encrypt them.

## 2.1 Key Generation

We derive the player's keys thanks to two arguments : the AACS password and the node id relative to the key. First we need to generate an AACS key from the AACS password. We used the function `CreateAESKeyMaterial`, provided in `KeyTree`. Then we created our own function `generateKey` that creates the player key. We apply MD5 to the result of xoring the node id and the AACS key. We use MD5 because it is convenient to create a 16 bytes key as we need. We know it is not safe but as the key itself is supposed to be secret, it does not matter.

## 2.2 File Generation

After generating the keys, we encrypt each key and the corresponding node id with AES-128 in CTR mode. Then we generate a MAC of the encrypted keys with SHA256. We used in both cases `SecretKeySpec` in order to derive the keys from the user password but in the encryption case, we used the `CreateAESKeyMaterial` to have a sufficient key length.

As the the MAC length is 32 bytes, about $2^{128}$ computations are needed in order to have a probability of 50% to have a collision. We think it is good enough.

Concerning the encryption, we use AES in CTR mode. We decided to encrypt the initialization vector, because having it in clear would make the encryption more vulnerable. As we have encrypted it, even if an attacker gets to know the key, he will not be able to decrypt the message. We had to do it with ECB, which is not the best mode of AES, but as the size of the IV is short, there is not too much redundancy. For the key, we first hashed the password, it allows for convenient size. Then the usage of AES in CTR mode seems to us to be a very good choice for safety.

We were forced to add the initialization vector to the foreseen content of the file, because otherwise we would have used AES in ECB mode, which would not be safe enough, as ECB allows for redundancy in the encrypted text.

# 3 DVDManufacturer

# 4 DVDPlayer

## 4.1 Retrieving the keys

This part occurs in two steps : decryption and analyse of the keys, these actions are performed by `decryptKeys` and `generateKeys` respectively. Given the keyfile, we must decode it, check its integrity and retrieve the keys in a convenient format. We know how the file is formatted and thus we can identify the bytes of the file corresponding to the initialization vector, the keys and the mac. The first thing to chek is the integrity of the document. If the MAC matches, then we decode the content thanks to the password and the initialization vector and return it.

As we have an upper bound on the complexity of the decryption of $K_t$, we decided that we would put the node id and the corresponding key of each node in a HashMap. Again as we know the format of the encoded content, we can retrieve the node id and the key, only on indexes basis.