Published in I.K. Sethi and R.C. Jain, Eds., *Storage and Retrieval for Image and Video Databases V*, **vol. 3022 of** *Proc. SPIE*, **pp. 290–297, SPIE, 1997.**

Practical solution to authentication of images with a secure camera

J.-J. Quisquater, B. Macq, M. Joye, N. Degand, and A. Bernard

UCL Crypto Group - Université catholique de Louvain
Place du Levant 3, B-1348 Louvain-la-Neuve, Belgium

URL: `http://www.dice.ucl.ac.be/crypto/`

ABSTRACT

Those who have seen the movie *Forrest Gump* witnessed former U.S. President J.F. Kennedy shaking the hand of Tom Hanks. Nevertheless, they have never met. Though this image seems real, how can we be certain it is not? How long can we still believe the images?

This paper presents an efficient system guaranteeing to detect whether the images of an original video sequence have been modified between the original recording and the moment of viewing. Moreover, it takes into account the intervention of an editor between the original recording and the projection. Whatever is the final edited tape, the cryptographic information generated by the camera allows to authenticate the images of the editing. This scheme is dedicated to images coming from digital cameras using the DV format but can be extended to any other standard.

Keywords: cryptography, authentication, images, secure camera, editing

1 INTRODUCTION

The communications of the next century will essentially be based on the images and it is time to wonder about the confidence we can give to them. The purpose of this paper is to find a way to prove whether a video image has been modified. By understanding the nature of the image, we will be able to distinguish reality from fiction. This will prevent future viewers from being mislead, as they will have the tools to identify the origin of the images and to appreciate the image by what it is actually.

This lead us to develop a system ensuring that a video sequence has not been modified between the original recording and the moment of viewing. For this purpose, we use a *secure* camera. It means that the camera holds a secret initialized by an external entity, often called Trusted Third Party (TTP). Nobody can get access to this secret. The camera authenticates each filmed image and the succession of images with its own secret.

Moreover, we take care about the possible intervention of an *editor* between recording and projection. We want to give the editor the means to authenticate his editing, using only the cryptographic information generated by the camera at the moment of the original recording. In other words, the editor transfers some cryptographic

information to the edited tape but does not generate himself cryptographic information.

Our solution allows to detect that any manipulation (e.g. modifications inside an image, permutation of two images or sequences of images) after the original recording. So, no valid signature can be generated outside the camera. This is why the editor can just transfer the cryptographic information but can not generate it. The signature scheme used is the Guillou-Quisquater [4] protocol.

The Guillou-Quisquater protocol (GQ) uses hash functions which are very sensitive to any modification of even one bit. Hence, we have to make sure that transmission errors would not prevent authenticating either pieces of editing or the original recording. By dividing the images in smaller entities, called blocks, and by developing a cryptographic error correcting code for each block, we have considerably reduced the probability to not authenticate an image due to an error of transmission (from 63% to 0.40%).

The remaining of this paper is organized as follows. We introduce in Sections 2 and 3 the tools that will be used and the basic functional model of the problem. We cut the images into blocks in Section 4 and we explain in Section 5 how we can sign the images and verify these signatures. In Section 6, we present some improvements that can be done to the speed processing. Finally, we conclude in Section 8.

2 TOOLS USED IN THE SOLUTION

2.1 Signature scheme

By signature, we mean a method to prove that the *integrity* of the message has been preserved and that the emitter of the message can be recovered. In order to sign a message, the signer uses his own secret which involves him in the signature process.

The GQ signature scheme is a "zero-knowledge" protocol, which means that the signer can prove to the verifier that he owns his secret without giving any information (knowledge) about it. In our case, the signer is the camera. It is not worth to explain completely the GQ signature scheme which requires some knowledge of cryptography and finite field theory. Nevertheless, we can easily summarize the philosophy of the protocol, after explaining what is a *one-way hash function*.

A hash function $\mathcal{H}$ applies on a message M of arbitrary length and the result $h = \mathcal{H}(M)$, called *digest* of the message, has a fixed length. Furthermore, the function is one-way. That means that given M, one can easily compute h, but the computation of M from h is infeasible.* Finally, for cryptographic purposes, the hash-function must be *resistant*. Finding two messages M and M' that are hashed to the same value h is infeasible. All these requirements are, for example, fulfilled by the US Standard SHA [1](Secure Hash Algorithm).

So far, we can see that we can use the hash function as a *data compressor* as well as a *digital fingerprint* of the message. In other words, we will sign the fingerprint of the image instead of the image itself, which decreases considerably the amount of calculus. We can also see that the hash functions spread out the errors: if M changes of even one bit, its digest will be totally different.

Now, we can describe briefly the philosophy of any public-key signature scheme. We call Susan the signer of the message and Victor the verifier.

1. Susan computes the digest of the message $h = \mathcal{H}(M)$ and signs this digest with the secret signature function $\mathcal{S}$, i.e. she computes $s = \mathcal{S}(h)$.

*Infeasible means computationally impossible.

2. Susan sends the pair (M, s) to Victor.

3. Victor computes the inverse of the signature using the public verifying function $\mathcal{V} = \mathcal{S}^{-1}$, i.e. he computes $h' = \mathcal{V}(s)$. Then he computes the digest of the received message $h = \mathcal{H}(M)$.

4. If $h' = h$, then Victor concludes that the message has not been modified.

In our scheme, we have chosen respectively the SHA algorithm and the GQ protocol for the hashing function and the signature scheme, respectively. Indeed, SHA has been commonly adopted as a standard hash function. Whatever the length of the input message, it produces a digest of 160 bits. The GQ protocol has been chosen for several reasons: it is a zero-knowledge protocol, it minimizes the amount of computations, it allows to parameter the level of security and it can easily be extended to a multisignature scheme.

2.2 Format of recording

The format of images we are working on is the numerical Digital Video [8] (DV) standard. This relatively new format is worldwide spread and has been adopted by all the common camera manufacturers. Let us first explain some useful characteristics of the format for our purpose.

The incoming light is divided into three basic colors using a prism. Hence, a color camera has 3 CCD (Charged Coupled Device). Each CCD, consisting of 410000 pixels, captures its color and translates it into electric signals. The three signals go then through a digital camera processor to obtain one signal of luminance (Y) and two signals of chrominance (R-Y) and (B-Y).

The luminance is sampled at 13.5 MHz while the chrominances are sampled at 3.375 MHz (in the NTSC format). This format is therefore also called 4:1:1. The quantization is 8 bits deep (256 levels). The total flow for one image is then 162 Mbps. A compression rate of 5:1 is achieved inside the camera using DCT and CLV algorithms reducing the flow down to 25 Mbps. The compression is intra-frame, similar to JPEG. In particular, this means there is no temporal redundancy compression. Each image is compressed individually. This allows to perform a very precise editing.

Furthermore, in the Sub Code Sector of the tracks of the tape, the camera writes the TIME CODE of the frame. This means that each frame is numbered as follows $H : MM : SS : n$ where n is the number of the frame within the second. This information will be used later.

The DV format includes another sector in which we can write up to 3.5 Mbps of auxiliary data, in addition to the video, audio and tracking signals. This is the location where the cryptographic information needed for the later authentication will be written.

Finally, the DV format uses errors correcting codes to reduce the number of transmission and decoding errors. However, it subsists a non-zero probability that one bit is incorrectly transmitted or decoded. We will assume this probability to be equal to 10^{-6}.

3 BASIC FUNCTIONAL MODEL

We can define five actors in our model: the secure camera, the editor, the Trusted Third Party (TTP), the verifier, the pirate.

- The *camera* signs the images and their succession during the recording in order to allow later authentication. It holds a secret initialized by the *TTP*. This secret involves the camera in the signature process.
- The *editor* makes an edited film by selecting some images of some sequences from the original film (while respecting the order of recording, i.e. he may not permute images or sequences of images). He transfers the needed cryptographic information in order to authenticate the edited film without using the original film.
- The *TTP* initializes each camera with a secret and owns a list of all secure cameras issued.
- The *verifier* checks the authenticity of the film, i.e. of the images separately and of their succession.
- The *pirate* tries to break or to weaken the scheme by any means in order to authenticate images which were not actually filmed. His aim is not to prevent us to authenticate films but to try to authenticate corrupted films.

4 DIVIDING IMAGES INTO BLOCKS

At a rate of 25 images per second, an image is coded on 25 Mbps/25 s^{-1} = 10^6 bits. Also, we know that a probability 10^{-6} of transmission error subsists even after the error correcting codes of the camera, due to the quality of the tape and of the VCR. Hence, the probability to have at least one incorrectly transmitted bit in a whole image is equal to 63%. On the other hand, we have seen that the hash function, which is the preliminary step in our signature scheme, spreads out the errors. So, we have a probability of 63% to refuse to authenticate an image only because of error of transmission, which is totally unacceptable. To overcome this drawback, each (compressed) image will be divided into 2^n blocks.

The whole image is error-free only if all the blocks are correctly transmitted, i.e. if all the bits of the image are correctly decoded. So far, dividing into blocks does not improve. However, if we define an additional Error Correcting Code (ECC) to correct up to one transmission error for each block, then the probability to have an error-free image is given thanks to the Bernouilli formula: [6]

$$\Pr(\text{error-free}) = \left((1-p)^{m-1}[(m-2)p+1]\right)^{2^n},$$

where 2^n is the number of blocks in which we divide the whole image, $m = 10^6/2^n$ is the size (in bits) of a block, and $p = 10^{-6}$ is the probability to have a transmission error. So, we obtain the following table.

n	2^n	m (in bits)	Pr(error-free)
0	1	1000000	73.58%
1	2	500000	82.77%
2	4	250000	89.81%
3	8	125000	94.39%
4	16	62500	97.04%
5	32	31250	98.48%
6	64	15625	99.22%
7	128	7813	99.60%
8	256	3907	99.78%

Table 1: Probability to have an error-free image of 2^n blocks using ECC algorithm.

The ECC used to correct one bit in each block works using the digest of the corresponding block. Indeed, for each block, the camera computes the digest $h = \text{SHA(Block)}$ and writes it on the auxiliary sector of the tape. If

there is one incorrectly transmitted bit within the block, then the digest of the received block $h' = \mathrm{SHA}(\mathrm{Block}')$ is different and we detect the error. To correct this error, we can switch successively each bit of the received block and computes again its digest until the two values h and h' correspond. Using this trick, the probability having a transmission error decreases seriously.

We still have to determine the number of the blocks. The more blocks we take, the less the transmission error is significant and the least amount of switching and computing of new hash values have to be made. On the other hand, the digest (160 bits using SHA) of each block will be written on the tape, and the more the number of blocks is large, the more the computation of the global hash value of an image (see Section 5) is time-consuming. A good trade-off is to divide the images into $2^7 = 128$ blocks. That leads to an average block size of 7813 bits and to a probability to not authenticate an image because of transmission error of 0.004.

5 AUTHENTICATION - EDITING - VERIFICATION

5.1 Authentication

Each compressed image i is divided into 128 blocks: $B_{i,1}, B_{i,2}, \ldots, B_{i,128}$ and to each block $B_{i,j}$ is associated a digest $h_{i,j} = \mathrm{SHA}(B_{i,j})$ that is written in the auxiliary data sector of the tape. A sequence consisting of images r to s will be denoted by $\Sigma_{r\to s}$.

We will not authenticate each block. The smallest entities we will authenticate using the signature scheme is the global hash value I_i of the image i which is defined as the hash result of the concatenation of the digests of all the blocks contained in the image, tied with the number of the image[†]

$$I_i = \mathrm{SHA}(h_{i,1}|h_{i,2}|\ldots|h_{i,128}, i).$$

In order to be able to authenticate the images and their position, we will not sign directly each image but the *succession witness* until image i which is defined recursively by

$$Z_i = \mathrm{SHA}(Z_{i-1}, I_i) \quad \text{and} \quad Z_0 = \emptyset.$$

This succession witness will then go through the GQ signature scheme. Call $s_i = \mathrm{GQ}(Z_i)$ $(1 \le i \le k)$ the corresponding GQ signature. This signature is written in the auxiliary data sector of the tape.

Remark. All the data written in the auxiliary data sector of the tape ($h_{i,j}$ and s_i) are not "protected" by the camera. We protect them using an Hamming error correcting code. Since the size of $h_{i,j}$ and s_i are relatively small, the probability to have an error is negligible.

5.2 Editing

The editor selects some sequences of images from the original film $\Sigma_{1\to k}$ consisting of images 1 to k while always ensuring the later authentication of the edited film.

Suppose the sequence $\Sigma_{\ell\to\ell+m}$ is removed. Then, in order to authenticate the next sequence, the verifier needs $Z_{\ell+m}$ to compute $Z_{\ell+m+1}, Z_{\ell+m+2}, \ldots$ Therefore, when the editor removes a sequence, he has just to write the succession witness of the last image he has removed, $Z_{\ell+m}$.

[†]The number i of an image can be obtained from the TIME CODE generated by the camera.

After the editing, the final film consists of a succession of sequences $\Sigma_{p\to q}$ ($p \leq q$) where the equality holds for an isolated image. For each sequence $\Sigma_{p\to q}$ kept, the editor copies on the tape in addition to the image itself the following elements:

- the digest of all the blocks of each image, i.e. $h_{i,j}$ for $p \leq i \leq q$ and $1 \leq j \leq 128$;
- the GQ signature of the succession witness until the last image of the sequence, i.e. $s_q = \mathrm{GQ}(Z_q)$. That allows the verifier to authenticate the whole sequence $\Sigma_{p\to q}$ and the succession of the images within the sequence.

Moreover, for each sequence $\Sigma_{\ell\to\ell+m}$ he has removed from the original recording, the editor writes the succession witness $Z_{\ell+m}$ of the last image of the removed sequence in order to reinitialize the succession witness computation for the next sequence.

Remarks. 1) For each sequence $\Sigma_{p\to q}$ he kept, we can imagine that the editor writes some intermediate GQ signature s_i, for example every 10 images. That bounds the probability to have a transmission/decoding error within the sequence. Moreover, that enables to more precisely locate the corrupted images, and then to authenticate the remaining of the sequence.
2) The digests $h_{i,j}$ are given to reduce the probability of transmission/decoding errors. However, there is another advantage: they allow the editor to modify one or several blocks inside an image without loosing the authentication of the remaining of the image. That enables, for instance, to insert a logo or a mask in the image.

5.3 Verification

To authenticate the film, the verifier has to check each image taken separately and their succession as follows.

1. For each block $B_{i,j}$, he computes $h'_{i,j} = \mathrm{SHA}(B_{i,j})$ and reads the data $h_{i,j}$ on the tape. If they are not equal, then he tries to correct one bit of the block using the ECC algorithm, as explained in Section 4. If they are still different, then the corresponding block will not be authenticated.
2. For each image Σ_i, he computes the global hash value $I'_i = \mathrm{SHA}(h_{i,1}|\ldots|h_{i,128}, i)$, where the values $h_{i,j}$ and i are read on the tape.
3. For each continuous sequence $\Sigma_{p\to q}$, he computes recursively $Z'_i = \mathrm{SHA}(Z'_{i-1}, I'_i)$ until $i = q$, where the initial value $Z'_{p-1} = Z_{p-1}$ is read on the tape.
   Then, he checks whether the GQ signature of the succession witness until image q is valid according to the computed value Z'_q, i.e. he checks whether $\mathrm{GQ}(Z'_q) = s_q$, where s_q is read on the tape. If yes, then all the images of the sequence and their position are authenticated by simply verifying one signature.
4. Finally, to verify the relative position of two consecutive continuous sequences, $\Sigma_{p\to q}$ and $\Sigma_{\hat{p}\to\hat{q}}$, the verifier has to check that the TIME CODE of the first image of $\Sigma_{\hat{p}\to\hat{q}}$ is greater than the TIME CODE of the last image of $\Sigma_{p\to q}$, i.e. he checks whether $\hat{p} > q$. Indeed, it is not possible to modify the image number i because this number is tied to the content of the image when I_i is computed.

6 SOME POSSIBLE IMPROVEMENTS

In this section, we will discuss some additional properties [6] that our system could cover if a new generation of more powerful processor and DSP appears.

First of all, we have never talked about the cameraman. It could be useful that the cameraman also is involved in the signature process such that each authenticated film would have a paternity. For this purpose, we just have to adapt the GQ signature into a multisignature [4] scheme. The cameraman would be represented by his personal smart card which keeps his secret, computes his part of the signature and communicates with the camera. However, it decreases the speed of the algorithm and has not been kept.

In some cases, the editor would like to modify the order of the original recording and to still authenticating *his* work. A simple solution is to re-compute all the succession witnesses and the corresponding signatures. Another solution is to use the Tillich-Zémor [9] hash function when computing the succession witnesses. They have then not to be re-computed.

Finally, powerful processors enables to use larger cryptographic parameters, which enhance the security of our system.

7 CONCLUSIONS

A system for authenticating images and their succession was developed. The two main advantages of our solution are 1) the authentication can still be done if an editor has removed some images from the original recording, and 2) the verifier doesn't need the original recording in order to authenticate the film. Also, thanks to our system only one signature is needed to authenticate a whole sequence of images. So, we reduced the problem of authenticating a sequence of images to the problem of verifying only one signature. Moreover, with this single signature, we can also authenticate the order of the images within a sequence. The posteriority of a sequence with another one can also be checked. Finally, our model allows the editor to modify one or several blocks of an image without keeping us authenticating the remaining of the sequence.

The main purpose of all these verifications may not be clear when we think about a movie. In a journalistic or legal context, it can become extremely important to give us the means to prove that this images, reports or interviews have actually been filmed, in a given order.

Notice that our scheme does not take into account the problem of the sound.

8 REFERENCES

[1] FIPS 180-1, "Secure Hash Standard," NIST, US Department of Commerce, Washington D.C., Apr. 1995.

[2] Bernard, A., and Degand, N., *Images authentifiées par caméra sécurisée*, Mémoire de fin d'études, Université catholique de Louvain, Louvain-la-Neuve, Belgium, June 1996.

[3] Guillou, L. C., and Quisquater, J.-J., "A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory," in *Advance in Cryptology – EUROCRYPT '88* (1988), C. G. Günther, Ed., vol. 330 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 123-128.

[4] Guillou, L. C., and Quisquater, J.-J., "A 'paradoxical' identity-based signature scheme resulting from zero-knowledge," in *Advance in Cryptology – CRYPTO '88* (1990), S. Goldwasser, Ed., vol. 403 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 216-231.

[5] Guillou, L. C., Ugon, M., and Quisquater, J.-J. "The smard card: A standardized security device dedicated to public cryptology," In *Contemporary cryptology - The science of information integrity* (1992), G. J. Simmons, Ed., IEEE Press, pp. 561-613.

[6] Joye, M., and Quisquater, J.-J., "Efficient authentication of sequences with the SL_2 hash function: Application to video sequences," Technical Report CG-1996/8, UCL Crypto Group, Belgium, Sept. 1996.

[7] Kelsey, J., Schneier, B., and Hall, C., "An authenticated camera," in *Proceedings of 12th Annual Computer Security Applications Conference*, ACM Press, December 1996, to appear.

[8] Sony, *The digital video handbook*, 1995.

[9] Tillich, J.-P., and Zémor, G., "Hashing with SL_2," in *Advance in Cryptology – CRYPTO '94* (1994), Y. Desmedt, Ed., vol. 839 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 40-49.