Watermarks and Text Transformations in Visual Document Authentication

Igor Fischer
ClickandBuy Services AG, Im Mediapark 5, Cologne, Germany
Email: igor.fischer@clickandbuy.com

Thorsten Herfet
Telecommunications Lab, Saarland University, Saarbrücken, Germany
Email: thorsten.herfet@nt.uni-saarland.de

***Abstract*— Integrity of digital documents is a very important issue if they are to be legally binding. Common solutions, like digital signatures or message authentication codes, are based on cryptography and require computers or similar hardware to be produced. They can be trusted only as long as the employed hardware can be trusted. We present a complementary scheme for document authentication in untrusted environment. The scheme combines visual authentication, human-recognizable watermarks, and image transformations, and is suitable also for larger text document. Because it relies on visual cryptography, it requires practically no computational power on the receiver side. To prevent potential attackers from obtaining signatures without author's knowledge, we propose using a simple challenge-response protocol.**

***Index Terms*— visual authentication, digital documents, watermarks, text deformations, CAPTCHA**

I. Introduction

Today, in the industrialized world, most documents are produced electronically, either completely or in large part. Not only is the production process more comfortable, but handling electronic documents—editing, storing, retrieving, searching, disseminating etc.— is much easier than for paper-based documents. However, paperless office, although envisioned decades ago, has not yet come to exist. For documents deemed "important", the hardcopy is still the prevalent medium. Although the notion of "importance" might vary from person to person, the consensus seems to be that documents of legal relevance are important. Simply put, any legally relevant document worth writing down is worth writing it down on paper, in non-electronic form. Let us look at some examples where both electronic and hardcopy documents are used:

- Many banks today offer online banking, which is faster, cheaper, and more convenient than traditional hand-filling of transfer forms or writing checks. Even the forms for opening an account can typically be downloaded over the Internet. However, to open an account, one will usually have to print the forms, hand-sign them and return them to the bank.
- Some countries (e.g. the United States) have visa applications forms available on the Internet. They can be filled out through a browser and a PDF file is automatically produced. Nevertheless, in order to apply for a visa, one has to print the file, attach his or her photograph, sign the form and send it to the consulate.
- Real estate can be found on online market places and acquisition negotiated per e-mail. But, the legal transfer of ownership is again done using printed and hand-signed contracts.

What these examples have in common is the signature, which—traditionally—serves as authentication. Signing a document is a willful act and one can safely assume that the signer has read the document before signing it. Also, signature can be traced back to its originator (or at least legal practice assumes so), so the signer cannot later deny his or her knowledge of, and consent with the document. Therefore, if a document carries the signature known to belong to a certain person, the following is assumed to hold:

1) The person whose signature appears on the document, an no-one else, gave its consent to the document, and
2) The person gave his or her consent exactly to the document, and not to anything else.

These properties are very useful in the case of a dispute. That the assumptions are not always valid is beyond the scope of this paper.

For digital documents, the same features should be provided by digital signatures [1]. User digitally signs a document by applying a cryptographic function to the document and a large number—the "private key"—which is user-specific and which he or she never discloses to anyone. As long as the key is kept private, the first of the above assumptions holds also for digital signatures. As far as the second assumption is concerned, digital signatures are actually more secure than hand-written ones, since they become invalid as soon as one bit in the document is modified. For paper-based documents, such minor forgeries might pass unnoticed.

This paper is based on "Visual Document Authentication Using Human-Recognizable Watermarks" by I. Fischer and Th. Herfet which appeared in the Proceedings of ETRICS 2006, Springer-Verlag LNCS 3995 © 2006 Springer-Verlag. In this paper a more formal presentation of the protocols is provided, new experimental results discussed, and an improvement proposed.

What has been tacitly assumed here is that the user performs the cryptographic computation himself. Due to computational complexity, this assumption is not realistic. In practice, the user will delegate the computation to a computer or similar cryptographic device. The problem which arises here is not only whether the device is trustworthy, but also whether the whole path, from the user to the device and back, is not compromised. Even for trusted devices (e.g. smart cards provided by a trusted authority), the document can be tampered with on its way to the device and back. If the device lacks sufficient displaying capabilities, as is typically the case for smart cards, the user cannot know what exactly the device signs in his or her name.

A. Attack Scenario

To better understand the danger, consider a typical scenario:

Alice[1] (a human) is writing a document (e.g. a contract with Bob) on her ordinary, general purpose computer. In addition, Alice can count on help from a trusted cryptographic device, which we call "Trent". He can be implemented, for instance, as a smart card and a reader attached to the computer. Trent is the keeper of Alice's signature and, upon request by Alice, signs documents he receives from her. Alice, having written the document, authenticates herself towards Trent (e.g. by typing in her PIN) and clicks the "Sign!" button on the computer. The computer sends the document to Trent, who digitally signs it and sends the signature back to Alice's computer[2]. Alice sends the document with the accompanying signature per e-mail to Bob.

Unfortunately, Mallory has installed a malicious program on Alice's computer, which enables Mallory to modify documents as she wishes without Alice's knowledge. When Alice sends the document to Trent, Mallory's program intercepts it, changes it (or lets Mallory herself remotely change it), and passes it further to Trent. Trent, not being aware of Mallory, signs the modified document and sends this signature to Alice. Due to computational complexity, Alice cannot check the signature herself and see that it actually does not fit to her document. She trusts the signature because she trusts Trent. When she sends the document and the signature to Bob, Mallory's program acts again in the background and substitutes Alice's document with the Mallory's. Consequently, Bob receives a document written by Mallory, which carries Alice's signature. The signature is perfectly valid; there is no way Bob can notice tampering.

So, although the digital signature algorithm might be perfect, the trusted device functioning properly, and user performing all actions as prescribed, this scheme fails nevertheless. It fails because the protocol involves a non-trustworthy participant: the general purpose computer. As long as computers are open platforms, where software can be freely installed, there is no way of completely preventing such scenarios.

Although *preventing* tampering with documents is impossible without completely trustworthy hardware, including the input and output interfaces, *detecting* tampering is not. Tampering occurs inside the computer, but its detection can be done from the outside. Detecting tampering might not be sufficient for Alice to have her work done: If she needs to send a signed contract to Bob, she still won't be able to do it. However, it might prevent greater damages which might occur if Bob received a forged contract. Once Alice notices that her document has been tampered with, she can stop using the compromised computer, instead of allowing it to send forged documents.

There are two basic approaches to ensuring that only authentic documents are signed and transmitted:

1) Hardware-based: Making the hardware trustworthy, so that tampering is either prevented or detected by the hardware, or
2) Human-based: Providing a mechanism which allows users to detect tampering and prevent further damage.

Attempts to pursue both approaches already exist, but have their limitations. An overview is given in the next two sections. In section IV we describe an approach belonging to the latter family, based on visual cryptography. Differing from previous approaches, it uses visually recognizable watermarks to ensure the authenticity of the message. Also, using a simple challenge-response protocol is proposed to prevent the attacker from obtaining valid signatures without authors consent. In section V we discuss a further improvement to the scheme, which makes it less dependent on the watermark resolution, texture, and quality.

II. Hardware-Based Approaches

The critical point in the above signing scenario is the path from the display to the trusted device. The user wishes to sign the document he or she sees on the display, but cannot be sure that this same document has arrived to the trusted device. The hardware-based approach is to make the display trustworthy[3].

A. External Trusted Device With Input and Output Capabilities

One possibility is to equip the external trusted hardware with the display. Such device could then present the received document to the user and ask his or her approval before producing the signature. For approval, there should also be an input interface, e.g. a small keyboard.

Similar approach is already in wide use for electronic payment, especially with debit cards. Readers for such cards include a small LCD and a numerical keypad and are certified by some trusted authority. In Germany, for

[1]The actors' names try to follow the convention from [2]

[2]Note that digital signatures need not to be incorporated in the document

[3]For completeness, we note that audible feedback, like synthesized reading of the text, might also serve the purpose, but would not be very practical.

example, the authority is the Central Credit Committee (*Zentraler Kreditausschuss*, ZKA), which lays down the criteria which card readers have to fulfill for a certain application. The highest security level is provided by Class 3 readers, with a built-in display and a keypad.

Although payment cards are generally not smart cards and the payment process is technically not digital signing, the above described readers could equally well be used in the signing scenario. With such readers, the user could verify the document on the display before using the keypad to initiate the signing. As long as the card reader and the card can be trusted—and this is the basic assumption behind the technology—this procedure is perfectly safe. In practice, however, due to physical limitations on the display, it would be useful only for very short documents: the reader's display is usually only a dozen or two characters wide with only a couple of lines. This is sufficient for displaying the price or merchant identification, but not for checking a legal document stretching over several pages.

Since in the above scenario the document is created outside the trusted device and displayed on it only for control purposes, this is only a tampering detection method. Prevention could be achieved only if the document could be produced directly on the trusted device. Although it is technically possible to produce large, high-resolution secure displays and full-size keyboards, the economic viability of such devices, with the sole purpose of facilitating digital signatures, is questionable.

B. Trusted Computing

Economic viability could be achieved through an opposite approach. Instead of adding human-accessible I/O capabilities to trusted device, the idea is to extend the trustworthiness to the whole general purpose computer, including its keyboard and display. This is the approach pursued by the Trusted Computing Group (TCG) and supporting technologies, like LaGrande or TrustZone.

TCG describes itself as a "not-for-profit organization formed to develop, define, and promote open standards for hardware-enabled trusted computing and security technologies, including hardware building blocks and software interfaces, across multiple platforms, peripherals, and devices" [3]. AMD, Hewlett-Packard, IBM, Intel Corporation, Microsoft and Sun Microsystems, Inc. act as promoters and the group has over a hundred participants.

As envisioned by the TCG, today's PC needs only a minor hardware modification in order to become a "trusted platform" (TP): A specialized, low-cost cryptographic microcontroller, called "Trusted Platform Module" (TPM), is added to the motherboard. Technically, TPM is similar to microcontrollers employed in smart cards. It consists of a processor, volatile and non-volatile memory, simple I/O bus, and a cryptographic coprocessor. Its main purpose is to produce digital signatures and provide safe storage for security-relevant data.

The platform relies on the TPM to trace the state and changes to its hardware and software, so no change can pass unnoticed. The trust in the platform is achieved through the chain of trust. At its beginning lies the first program which executes at power-on. In common personal computers it is typically stored in the BIOS. At power-on this program would "measure" the next program(s) to be executed, by computing a checksum or some other hash value of its or theirs code. This value is compared with the one stored and signed by the TPM. The initial values are stored by some trusted entity, e.g. the computer manufacturer. If the values match, it means that the status of the tested program(s) has not changed and that it or they can still be trusted. Thus the next program is executed and performs further "measurements" on the hardware (checking the graphics adapter, the hard disk etc.) and uses the same mechanism to check if the values are correct, i.e. that nothing has been changed. Again, if this test passes, the hardware can be trusted. This chain unfolds further, over the operating system loader and the operating system, up to application programs. In each step, a program checks its successor before executing it.

The security in this approach relies on the trustworthiness of the BIOS (the trust in TPM is given by definition). As long as the attacker cannot manipulate the first program executed after power-on, he or she cannot plant any program unnoticed, including, but not limited to malicious ones. Since the BIOS is typically stored in ROM, the attacker would need to have physical access to the computer in order to tamper with it. Even then, the computer might be physically protected, e.g. sealed, so that user could detect unauthorized opening. The Trusted Computing (TC) specification requires at least that the BIOS is physically marked so that removing it cannot pass unnoticed [4], at least for someone who bothers to take a look inside the computer. For even higher protection it is envisioned for the future to place the first program to be executed inside the TPM.

The low cost of TC hardware comes at a price. Since most of the security is outsourced to software, the software becomes increasingly complex. As a consequence, despite optimistic announcements, TC-enabled products are not yet available. Customers also seem to be reluctant about accepting it, as the technology has faced serious criticism. For example, the German Association of Insurance Industry (GDV) is decidedly against TC, for three basic reasons: lack of legal framework, lack of control possibility and high misuse potential [5]. It is feared, among other things, that through TC manufacturers might coerce users into using or not using some software or hardware, and that private information might be indirectly disclosed without user's knowledge. Also, it is not clear how backups would work and what happens in the case of a hardware failure. The TCG has attempted to dispel the fears [6], but their success remains unknown. That the fears are not baseless is indirectly confirmed by the TCG best-practice manual [7], which denounces such misuses of the TC technology. The manual is, however, only a recommendation, and compliance with it cannot be enforced. Experience shows that weaknesses tend to

be exploited.

Notice that TC in its current form enables cautious owner to detect manipulations on his or her computer. It does not differentiate between a "good" manipulation (for example, adding a new hard disk) and "bad" one (infection by a virus) and leaves the action decision to the owner. It also does not protect remote users (e.g. his or her communication partners) from willful abuse by the computer's owner. If the owner alters the BIOS, remote users have no way of knowing it. Therefore, to achieve the level of trust needed for legally binding documents, TC platforms will have to be complemented by technology which is unconditionally trusted, such as external smart cards.

III. Detecting Tampering Using Visual Cryptography

A. Visual Cryptography

Beside their technical drawbacks, the above approaches have to overcome the psychological barrier of users reluctant to buy and install new hardware. An appealing, low-tech alternative for short messages is visual authentication [8]. The idea was originally developed for authentication of electronic payments and is based on visual cryptography [9].

Visual cryptography is a perfectly secure cryptographic method based on a visual secret key. While the encryption is computationally intensive and is done by the computer, the decryption can be performed with little conscious effort by the human visual system. The method is most easily implemented for encrypting black-and-white images, and works as follows:

Each pixel in the original image is divided into a block of $N \times N$ so-called "subpixels". Technically, this can be done by scaling the image up by the factor N, simply by repeating each pixel $N \times N$ times, without any smoothing or interpolation. Thus, for each original pixel in the image, there will be a block of $N \times N$ subpixels. For an originally black pixel, they will all be black, and white for originally white pixels. In the next step, in every "white" $N \times N$ block (corresponding to originally white pixels) a random half of the subpixels is flipped to black[4]. Such blocks, looked at from appropriate distance, visually appear gray. It can be said that the original black-and-white image has been transformed into another, which visually appears to be black-and-gray.

The final step is the actual "encryption". The transformed image is split into two half-images, so-called "shares", block-by-block. For understanding it is useful to consider white subpixels to be transparent, as if printed on a transparency. For practical implementation, one of the shares is actually printed on a transparency. Then, each "white" block is placed on both shares, so that overlaying them does not change block appearance. But for black blocks, two complementary half-blocks are produced, each with a different half of the subpixels black, and placed each on one share. An example is shown in Figure 1.

Each share is, visually and statistically, indistinguishable from a random pixel distribution. But, when laid one over another, the black-and-gray image reappears. If we denote black subpixels in the shares with 1 and white with 0, superimposing the shares corresponds to binary OR.

This is a visual implementation of the 2-out-of-2 secret sharing technique [10]. The basic idea of k-out-of-n secret sharing is to split a message into n "shares", so that neither of them alone, nor any combination of less than k of them, reveal anything about the message, but k shares combined are sufficient to reconstruct the whole message. If $n = k = 2$, this is actually a one-time pad cryptography. One of the shares (transparencies in our case) is used as the cyphertext and the other as the key. It is obvious that one of the shares—the key—can be fixed and agreed in advance, and the other one computed from it and the document to encrypt. In the visual cryptography case, this means producing a transparency with a random-looking pixel pattern in advance and giving a copy of it to the communication parties.

B. Visual Authentication

Visual cryptography can be used for authentication, although not directly. Basically, the idea is for the user (Alice) and the trusted device (Trent) to share a secret key—Alice would have it in the form of a pre-printed transparency, and Trent as a binary file. To allow Alice to detect tampering, Trent would visually encrypt the received document and send it back to Alice. She would visually check if it is identical to the document she sent him and only if yes, use the signed document. This simple approach, as described in Protocol 1, however, won't work well, because Mallory can be assumed to know the

Protocol 1 A naïve, non-working attempt to perform visual authentication.

Require: Trent and Alice share a visual secret key K. Trent can be unconditionally trusted.

Alice:
Produce a digital text document D.
$B \leftarrow$ bitmapImage(D).
Send B to Trent.

Trent:
Receive B' (possibly identical to B).
$E \leftarrow$ visuallyEncrypt(B', K).
Send E to Alice for authentication.

Alice:
$D'' \leftarrow$ visuallyDecrypt(E, K).
if D'' semantically equals D **then**
  Consider D' authentic.
end if

[4]For this reason N is usually chosen as a multiple of 2, otherwise there would be an unequal number of black and white subpixels in a block

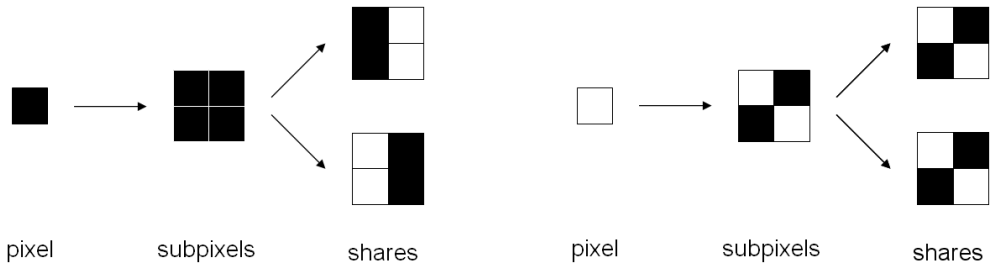

Figure 1. An example of splitting a black (left) and a white (right) pixel in two shares for visual cryptography. Clearly, other choices of subpixels are possible.

content of the document D or can deduce it by applying character recognition on its bitmap image B, which she intercepts. If she also intercepts the encrypted image E, she can deduce the secret key K. Knowing the key, she can produce any document and properly encrypt it, and Alice would believe it comes from Trent.

Things get only slightly better if Alice sends the ASCII text to Trent, and he produces the bitmap, as in Protocol 2. Here, Mallory also needs to know the parameters (font, size, etc.) for producing the document image B before she can deduce the secret key K. Considering that the number of fonts and font sizes which come into consideration is quite limited, this is not particularly secure. Also, sending ASCII text is unlikely to be attractive for users who want their documents to be formatted. Sending their bit-mapped images is clearly an overkill, because the formatted document is produced using some document description language (RTF, PostScript, XML+FO, LaTeXcode, PDF...). If Trent would understand it, he could produce the image of the formatted document. Unfortunately, the formatting would then also be available to Mallory, so that the Protocol 2 is not applicable.

Protocol 2 Slightly improved naïve attempt to perform visual authentication.

Require: Trent and Alice share a visual secret key K. Trent can be unconditionally trusted.

Alice:
Produce a digital text document D.
Send D to Trent.

Trent:
Receive D' (possibly identical to D).
$B \leftarrow$ bitmapImage(D').
$E \leftarrow$ visuallyEncrypt(B, K).
Send E to Alice for authentication.

Alice:
$D'' \leftarrow$ visuallyDecrypt(E, K).
if D'' semantically equals D **then**
Consider D' authentic.
end if

To achieve security, Trent must expand the document image with information known to Alice, but not to Mallory, before encrypting it. Several related methods have been proposed in [8]:

1) Content/Black Areas: The transparency is twice as big as the document image and is divided into two areas, "black" and "content". Trent and Alice have agreed in advance which area is which, but Mallory does not know it. Trent constructs the cyphertext so that the document appears in the "content" area, and the "black" area is completely black. If Mallory tries to tamper with the document, she has $\frac{1}{2}$ chance of showing the document in the wrong area. Security can be further increased by having more $k > 2$ areas (and correspondingly bigger transparencies), where Mallory's chances fall to $\frac{1}{2^k-1}$. Also, the transparency has to be a one-time pad, otherwise the "black" area of the cyphertext would not change from message to message and Mallory could simply identify it.
2) Position on the Screen (or, generally, output device): This is in a sense a generalization of the first method. The transparency has a marked area in which the document has to appear and Mallory does not know where this area is positioned. Mallory chances fall with the difference between the area and the document size.
3) Black and Gray: instead of "white", "gray" is used. It is encoded by having three quarters of the subpixels in a block black and one quarter white. This increases the security even when the plaintext is known, because for a fixed share of a gray block there are many ways (four in case of $N = 2$) of constructing the other share. So, for every "gray" pixel in the original image, Mallory has only a low probability of turning it into black. It does not hold for the opposite direction, however, so Trent is required to send the document (black on gray) and its inverted version (gray on black) to Alice for checking. The other drawback is that this approach reduces the contrast, making the result difficult to visually recognize.

All above approaches require transparencies bigger than the document and possibly reduce its readability. These are not grave issues for the originally envisioned electronic payment application, where the documents would be short (like the price to pay) and displayed on a high-contrast screen. However, for documents consisting of several pages they might be impractical.

IV. Using Watermarks for Obfuscating the Plaintext

It is worth noticing that already visual *cryptography* considerably blows up the data needed for processing. Instead of encrypting simple text, the algorithms now have to work on images, which are much larger in terms of memory. Obviously, the images must carry a great deal of redundancies. Can we exploit them, instead of further enlarging the images, as the above visual *authentication* approaches require?

A method which does not physically enlarge the authentication image was presented in [11]. The idea is still to enrich the image with information unknown to Mallory, but, instead of placing them in extra areas, the information are embedded *in* the image. A separate, faint image, serving as watermark is incorporated into the document. The term "watermark" might be somewhat confusing, since in the digital world it is sometimes used for non-perceptible but machine-detectable data hidden in e.g. audio or image files. For the purpose of this text, watermarks are understood as digital, but visible, human-recognizable images—the meaning more closely resembling original, centuries-old, paper-based watermarks. The proposed method can be summarized as in Protocol 3. Since it does not require enlarging the image area, it is more suitable for larger text documents. Beside the secret key, as in above discussed methods, Alice and Trent share a watermark: a visual pattern to be incorporated in the document image. Like the key, Alice has the watermark as a physical object, for example printed on a piece of paper. This is necessary because, as in other methods, Alice's computer and its connections can be compromised. Trent is a machine—an external hardware device (a smart card, a USB stick...), a trusted remote computer (in Alice's LAN, connected over Bluetooth...), or a tamper resistant program [12] running on the user's PC—and, naturally, has the watermark in a digital form.

Protocol 3 Visual authentication using visual watermarks.

Require: Trent and Alice share a visual secret key K and a watermark W. Trent can be unconditionally trusted.

Alice:
Produce a digital text document D.
Send D to Trent.

Trent:
Receive D' (possibly identical to D).
$B \leftarrow \text{bitmapImage}(D' \vee W)$.
$E \leftarrow \text{visuallyEncrypt}(B, K)$.
Send E to Alice for authentication.

Alice:
if E looks like uniform distribution of pixels **then**
  $D'' \leftarrow \text{visuallyDecrypt}(E, K)$.
  if D'' semantically equals D **and** D'' contains W **then**
    Consider D' authentic.
  end if
end if

Once Alice is convinced that D' (the document received by Trent) is authentic, she can request Trent to digitally sign it and send her the signature. To prevent Mallory from impersonating Alice and sending the request for signature for a forged document, using a simple challenge-response protocol has been proposed in [11]. In addition to key K and watermark W, Alice and Trent share an authorization code C, for example a short (4-8 characters) alphanumeric string. Alice requests the signature by sending the code to Trent, as Protocol 4 shows.

Figure 2 visualizes the workflow and shows possible attack points for Mallory. To prevent replay attacks, each combination (W, C) should be applied only once. As a consequence, for signing multiple documents Alice and Trent need a whole list of watermarks and codes. Such a list is just a more sophisticated variant of a Transaction Authorization Number (TAN) list, often used in Europe for online banking. It can be produced and distributed in a similar way as the TAN lists, by the entity which manufactures or distributes Trent. Alice could get her list per post and Trent electronically and in encrypted form.

Protocol 4 Authorizing the signing process.

Require: Trent and Alice share a visual secret key K, a watermark W, and a short alphanumeric code C. Trent can be unconditionally trusted.

Alice:
Run Protocol 3 to produce document D and check if it has reached Trent unmodified.
if Trent has received authentic D **then**
  Send C to Trent.
end if

Trent:
Receive C' (possibly identical to C).
if $C' = C$ **then**
  $S \leftarrow \text{sign}(D)$.
  Send S to Alice.
end if

An example for the document, watermark, transparency (key), encrypted document (produced by Trent), and the superposition result is shown in Figure 3.

A. Attacks on Watermarked Documents

Watermarking the document is a process of binary ORing the two (here we use the convention black=1 and white=0). Consequently, black pixels remain unchanged, only white pixels in the document can by ORing become black. Since Mallory knows the document and the encrypted watermarked image, she can deduce how the black pixels in the document (but not in the watermarked image!) are encoded, i.e. split into shares. But, this would not be enough to allow her to meaningfully modify it.

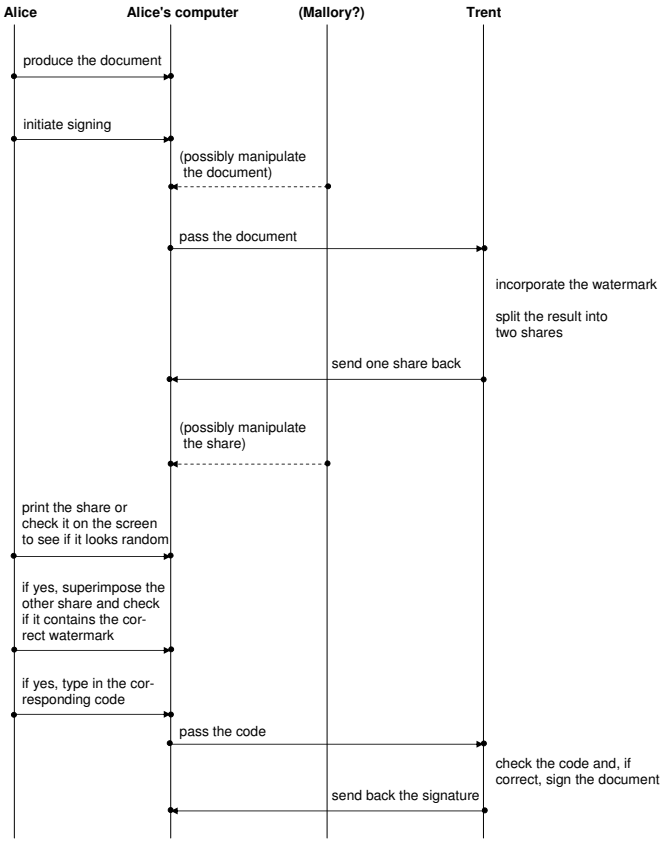

Figure 2. Document authentication protocol with possible attack points for Mallory

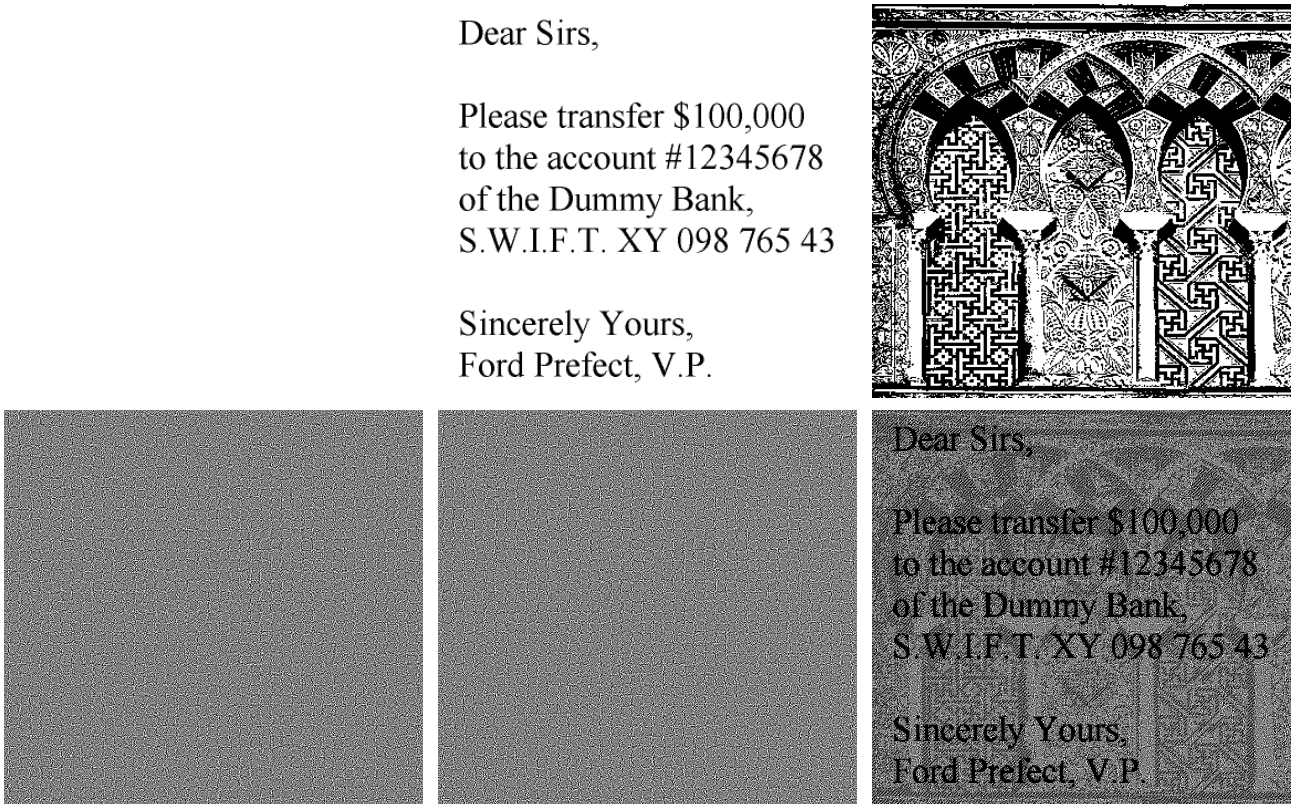

Figure 3. An example of visual authentication through watermarking. Top row: the image of the original document (middle) and the watermark image (right). Bottom row, from left to right: the share agreed in advance to serve as the secret key (transparency for the human user); the share produced by the signing device ("Trent"); and the watermarked and encoded document. Each share alone looks like a uniform distribution of black and white pixels, but overlaid they produce recognizable document with the watermark.

She would need to be able to turn black document pixels into white, and white into black, without damaging the watermark.

Consider the task of turning a black document pixel $d = 1$ into white. Through watermarking the pixel is ORed with the corresponding watermark pixel w, which is unknown to Mallory. She knows the superposition result, $d \vee w = 1$ and how it is coded in shares, but, not knowing w, cannot deduce $\overline{d} \vee w$. She has two options:

1) She can force the corresponding block to "white", by inverting its share, or
2) flip a coin and decide whether to leave it as it is or invert it.

In both cases she is guessing the value of w and her chances depend on the distribution of white and black in the watermark. For watermarks with an equal number of black and white pixels Mallory's chances of guessing one random pixel are $1/2$.

In turning a white document pixel into black, Mallory has similar problems. She knows $d = 0$, but not whether $d \vee w$ is 0 or 1. If she decides to invert the corresponding share, she again runs into the risk of distorting the watermark. What she can do is to force an illegal share, by making all subpixels in the block black, but such tampering with the cyphertext is easily spotted before overlaying it with the secret key.

It was argued that in a typical document, where a character is composed of dozens of pixels, Mallory's chances of tampering with the document without distorting the watermark would be negligible. However, our experiments show that watermark is a very sensitive issue, where pixel distributions vary considerably across the image. Whether Mallory's chances are good or bad depends strongly on the watermark, the place, and scope of modification. An example is shown in Figure 4. Mallory tries to change the account number from #12345678 to #12345679—a minor change in terms of Hamming distance, but which might be very important semantically—by randomly flipping pixels. Because the watermark is quite "noisy" at the position where the manipulation takes place and because it is embedded only as a faint background image, the tampering is visually not easy to detect. Only a careful observer will notice that there is something wrong with the digit 9.

Figure 5 shows the other type of attack, where Mallory produces an illegal share by forcing desired pixels to be black. It is practically impossible to detect the forgery by looking on the watermarked image, but the share is clearly illegal.

For comparison, Figure 6 shows cases where Mallory makes a bigger modification on the document.

V. Text Deformations Increase Security

Attacks like the above are possible because Mallory knows exact position of every pixel she wishes to modify. A straightforward idea to prevent them is then to let Trent move the text around in the document before applying visual cryptography. The idea was proposed in [11] in the context of applying one key for more than one authentication. Here we extend it by allowing more visible text transformations, for the purpose of preventing attacks even in the case of weak watermarks.

The transformations can include translations, rotations, or even distortions. The distortions are not so big to prevent the user from recognizing the text, but can be big enough to prevent current computers. This is the principle behind visual CAPTCHAs [13], which many Internet services apply to block robots or agents from consuming the resources, while granting access to humans. CAPTCHAs can also be used for document authentication [14]. In the context of this paper, we use text transformations as an additional security feature in visual authentication. They carry no information for the user and need not be agreed in advance between him/her and Trent. The authentication protocol is almost identical to Protocol 3, the only difference being that Trent, after producing the bit-mapped image B of the document and before visually encrypting it employs a transformation T.

Figure 7 shows an example of such visually transformed document and the result from attack attempts. Notice that Mallory knows that the text is being transformed, and even knows which kind of transformation is used ("water ripples" in this example), but does not know all the necessary parameters.

At this point, we might ask ourselves if the watermarks are necessary when text transformations are applied? Couldn't we say that the transformation serves as a kind of "watermark"? In some cases, particularly for densely written text, this might be true. But notice that transformations transform only the text, but not the empty areas inside and outside of it. On the other hand, watermarks are most useful in empty areas, where they are not covered by text. Actually, watermarks and transformations are complementary.

VI. Conclusion

Numerical (as opposed to visual) cryptography offers a high level of protection for digital documents and is essential in ensuring an efficient and secure electronic communication. However, due to its computational complexity, users have to rely on machines (computers or specialized hardware) for applying it. A big challenge has been securing the path from the human to the cryptographic machine. This path is currently the weakest link, which limits the security of the whole cryptographic chain.

In this paper, a method for document authentication between the cryptographic module and the user was presented. The method utilizes visual cryptography, visual text transformations, and watermarking. It requires no additional computer hardware and is very easy to implement using existing infrastructure. This method compares favorably to previous similar methods, because it uses the shared secret area (the transparency) more efficiently. This makes it much more suitable for authentication of larger documents, like full-page contracts. The proposed

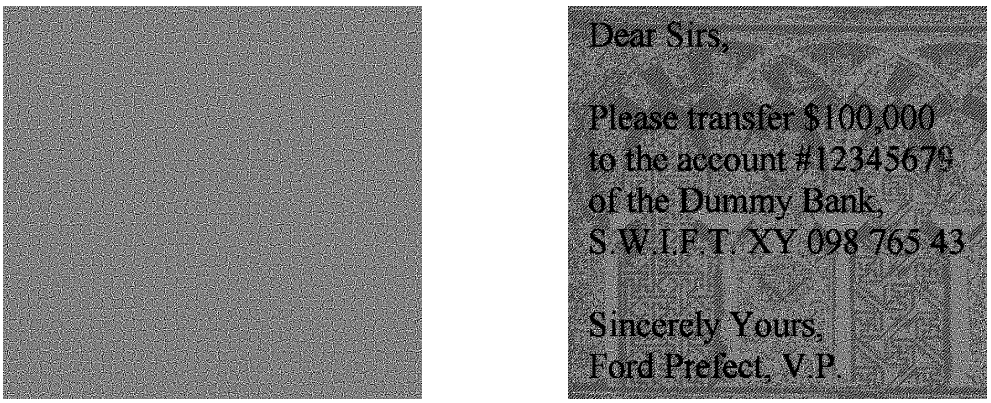

Figure 4. An example attack with randomly inverting shares of pixels that need to be manipulated. The forged document share (left) is visually indistinguishable from the original one and even in the superposition with the key the tampering is hard to see (right).

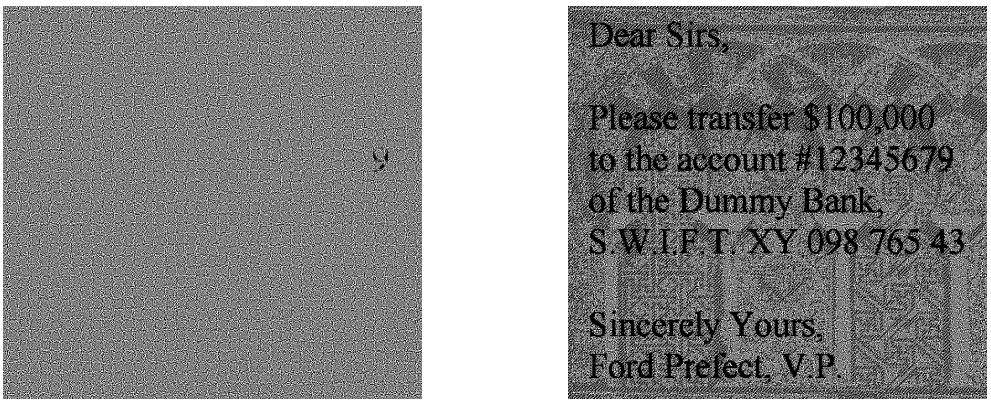

Figure 5. An example attack, where the attacker, for the pixels he wants black, forces all subpixels in the share to be black. The result (right) is almost indistinguishable from a legitimate document, but its encrypted share (left) is illegal, because the pixel distribution is not uniform. The digit 9 is almost completely recognizable in it.

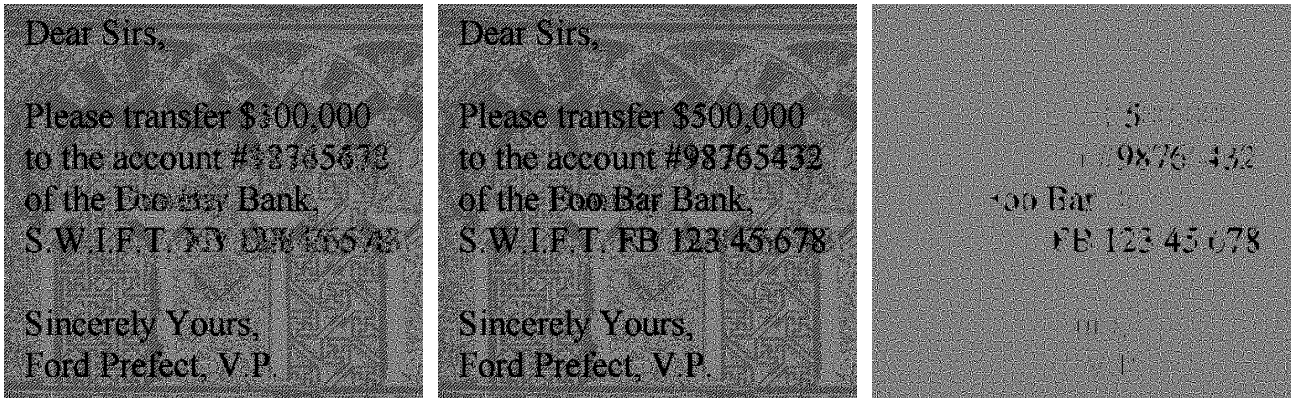

Figure 6. An attack example, where Mallory attempts large modifications on the document. **Left:** Result of a "random flipping attack". The text is hardly recognizable and tampering is obvious. **Middle:** Result of a "forced black" attack. The text looks much better and tampering is harder to spot. But: tampering becomes obvious if one takes a look on the document share produced by Mallory (**right**).

challenge-response mechanism prevents the man-in-the-middle attacker from obtaining a signed document without author's approval. Assuming that the author would not approve signing a forged document, the attacker is prevented from obtaining a valid signature for it.

Notice that the method is intended strictly for local use, between the user and his or her cryptographic module. For communication with remote users, classical numerical cryptography is still needed [15]. Also, the method is not intended for to be used among arbitrary number of users and trusted devices. It essentially relies on symmetric cryptography, so the number of key sets (staples of transparencies and watermarks) increases linearly with the number of user per trusted device. However, for the envisioned application—securing the channel between a user and her trusted device—there should be only one key set per user.

An implementation issue that needs to be clarified is key distribution and management. On the user's side, the method requires a list of watermarks and a staple of trans-

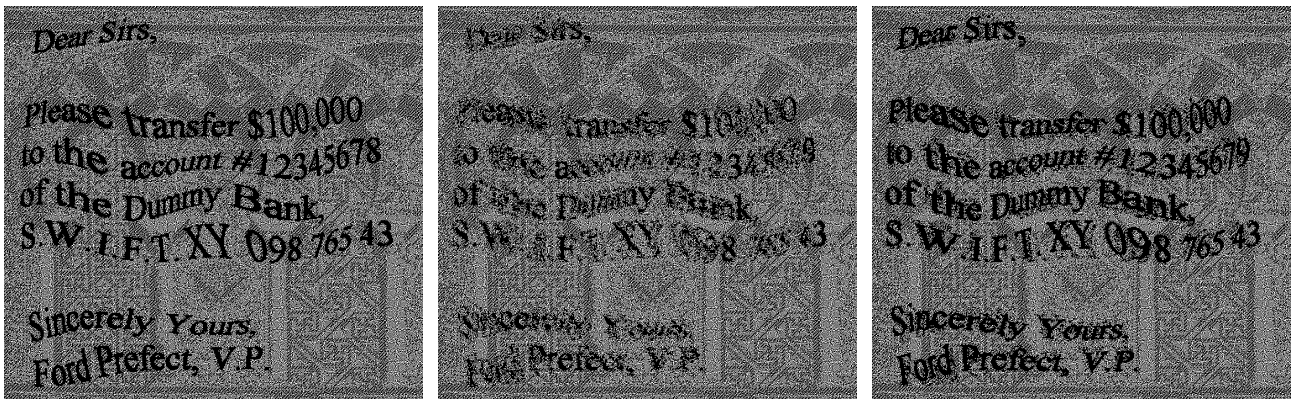

Figure 7. Example of visually transformed text and attacks on it. Mallory introduces only a minor modification, trying to change one digit in the account number. **Left:** Transformed and watermarked original document. **Middle:** Result of a "random flipping attack". The text is hardly recognizable and tampering is obvious. **Right:** Result of a "forced black" attack. The text looks better, but tampering is still visible. As in above examples, tampering becomes obvious if one takes a look on the document share produced by Mallory (not shown).

parencies. To achieve optimal security, a new watermark and a transparency are needed for authenticating each document page. The watermarks, which appear only faint over the document, can be printed in high contrast, but reduced in size, so that a dozen or two fit on a sheet of paper. The transparencies, however, have to be full-sized and would probably be distributed in a form of a booklet. They both must be distributed in a physical, and not electronic form. Otherwise, an authentication mechanism would have to be provided, which is a recursion of the problem we try to solve. The distributing authority would be electronically distribute the corresponding files to cryptographic modules. The security and confidentiality would be guaranteed by classical cryptography.

REFERENCES

[1] R. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, pp. 120–126, February 1978.

[2] B. Schneier, *Applied Cryptography*. John Wiley & Sons, Inc., 1996.

[3] Trusted Computing Group, "Home page," 2005. [Online]. Available: https://www.trustedcomputinggroup.org/about

[4] S. Pearson, Ed., *Trusted Computing Platforms*. Upper Saddle River, New Jersey: Prentice Hall PTR, 2003.

[5] F. Chiachiarella, U. Fasting, T. Fey, S. Leppler, G. Lux, P. Lubb, A. Moser, G. Otten, J. Schlattmann, S. Schumann, L. Schweizer, and F.-J. Souren, "Das Risiko Trusted Computing für die deutsche Versicherungswirtschaft," *Schriftenreihe des Betriebswirtschaftlichen Institutes des GDV*, vol. 13, 2004. [Online]. Available: http://www.gdv-online.de/tcg/pos_tcg.pdf

[6] Trusted Computing Group, "Trusted Computing Group Clarifications for the German Insurance Industry Association paper "The Threat, Trusted Computing, to the German Insurance Industry"," 2005. [Online]. Available: https://www.trustedcomputinggroup.org-/downloads/whitepapers/GDV_Clarification_from_TCG-_v8_English.pdf

[7] TCG Best Practices Committee, "Design, implementation, and usage principles for TPM-based platforms," 2005. [Online]. Available: https://www.trustedcomputinggroup-.org/downloads/bestpractices/Best_Practices_Principles-_Document_v1.0.pdf

[8] M. Naor and B. Pinkas, "Visual authentication and identification," in *CRYPTO '97: Proceedings of the 17th Annual International Cryptology Conference on Advances in Cryptology*. London, UK: Springer-Verlag, 1997, pp. 322–336.

[9] M. Naor and A. Shamir, "Visual cryptography," *Lecture Notes in Computer Science*, vol. 950, pp. 1–12, 1995. [Online]. Available: citeseer.ist.psu.edu/naor95visual.html

[10] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, November 1979.

[11] I. Fischer and T. Herfet, "Visual document authentication using human-recognizable watermarks," in *Proceedings of ETRICS 2006, LNCS 3995*. Springer-Verlag, June 2006, pp. 509–521.

[12] D. Aucsmith, "Tamper resistant software: An implementation," in *Proceedings of the First International Workshop on Information Hiding*. London, UK: Springer-Verlag, 1996, pp. 317–333.

[13] L. von Ahn, M. Blum, N. Hopper, and J. Langford, "CAPTCHA: Using hard AI problems for security," in *Proceedings of Eurocrypt*, 2003, pp. 294–311. [Online]. Available: citeseer.ist.psu.edu/vonahn03captcha.html

[14] I. Fischer and T. Herfet, "Visual CAPTCHAs for document authentication," in *Proceedings of the IEEE International Workshop on Multimedia Signal Processing (MMSP)*. IEEE, October 2006, pp. 471–474.

[15] ——, "Visually authenticated communication," in *Proceedings of International Symposium on System and Information Security*, J. P. de Oliviera, C. Westphall, and J. Brustoloni, Eds. CTA/ITA/IEC, Sã José dos Campos, Sao Paulo, Brazil, November 2006, pp. 471–474.

Igor Fischer received his Diploma in Electrical Engineering from the University in Zagreb, Croatia and his Ph.D. in Computer Science from the University of Tübingen, Germany. He has been a postdoctoral fellow at the Hebrew University in Jerusalem, Israel and worked as a researcher at the Saarland University, Germany. Currently he works for ClickandBuy. His past and current research interests include multimedia, machine learning, and security.

Thorsten Herfet was born in Bochum, Germany and received his Ph.D. in engineering in 1991. During 10 years in industrial research he e.g. held the position of VP Research within Grundig. Since 2004 he is the Chair of the Telecommunications Lab at Saarland University, Germany and currently also acts as the Dean. His research interest are centered around anylysis, pocessing and distribution of audio/visual content. He is Member of the IEEE.