Visually Authenticated Communication

Igor Fischer
Telecommunications Lab
Saarland University
66041 Saarbrücken, Germany
igor.fischer@nt.uni-saarland.de

Thorsten Herfet
Telecommunications Lab
Saarland University
66041 Saarbrücken, Germany
thorsten.herfet@nt.uni-saarland.de

***Abstract*— Visual document authentication is a method by which humans can check document authenticity directly, without relying on computers for cryptography. In order to be applicable, it is essential that the document is represented as an image, recognizable to humans. Representing text documents as images has a series of drawbacks, like high memory and bandwidth requirements. In this paper we present two protocols for electronic exchange of visually authenticable digital documents, one requiring high bandwidth, but easy to implement in current infrastructure, and the other with low bandwidth requirements and preservation of the original document.**

I. Introduction

Digital documents have a number of advantages compared to their hardcopy counterparts. They are easier to produce, store, retrieve, transport, copy, edit, search, and so on. On the other hand, they can be manipulated on the way from the author to the receiver or after reception, e.g. on receiver's computer. Since document authenticity is usually important, and even essential in case of legally binding documents, digital signatures [10] are often used for ensuring that such modifications do not pass unnoticed by the receiver. Digital signatures are cryptographic methods and, due to the computational complexity, the author has to rely on computer to for digitally signing a document. Document's signature is simply a binary file without an obvious meaning to a human observer, so for checking its validity the receiver will also have to use his computer. If the signature matches the document *and if both the author's and the receiver's computer, including whole cryptography involved (algorithms, keys etc.) can be trusted*, it can be safely assumed that the document has arrived unaltered. In other words, the question of trust has shifted: instead of having to blindly trust in the authenticity and the integrity of each document, the communication parties have to trust their computers to perform the involved computations correctly. This is an obvious progress, compared to not using digital signatures, since one trusted computer can usually be used for signing/checking of many documents. However, as soon as general purpose personal computers (PCs) are involved, even if only as an intermediary, this trust cannot be taken for granted. Being open platforms, they are susceptible to viruses, Trojan horses, worms and other hacker attacks, which can manipulate the document before digitally signing it. Using a trusted hardware attached to the PC to perform the cryptography, e.g. a smartcard, does not solve the problem, either. As long as the PC is vulnerable, the document can be tampered with inside it, on its way to the smartcard, and the user has no way of noticing it. A step-by-step description of an attack is given in the next section.

The danger has been recognized long ago and several approaches have been proposed to counter it. They are briefly reviewed in section III. Some of these approaches require an established hardware infrastructure currently not available. Others are remarkably simple, relying on visual authentication, but have the drawback that documents have to be represented as images. Transforming text documents into images induces several problems. First, images are much larger in terms of memory consumption and need significantly higher bandwidth for transmission. But, more importantly, images of text documents lack many of the advantages original digital documents have. While searching for words in a text document is trivial, doing it in an image is a complex task involving pattern recognition. Also, editing such a document is virtually impossible without transforming it back into a text format. While such transformation might be possible for images where text is clearly recognizable, the essence of visual authentication techniques is to additionally modify images, so that reconstructing the original text becomes hard, if not even impossible.

We therefore propose a document authentication protocol which combines visual authentication with classical cryptography. Visual authentication is used only locally, at author's and receiver's side, but the document itself is transmitted as text. The protocol is described in section V. Such protocol requires an infrastructure of trusted modules (hardware devices or software), which is currently not available. We therefore also describe a protocol which is easier to implement in current infrastructure.

II. Attack scenario

Consider a communication between standard cryptographic personae, Alice and Bob. The communication is not necessarily confidential, but its authenticity has to be guaranteed. Alice is writing a document, for example a contract with Bob, on her PC. Here, PC stands for any kind of personal computer, where the user is usually identical with the administrator and can freely install and remove hard- and software. Alice, being only a human, lacks the necessary computational power to digitally sign the document herself, i.e. perform the cryptographic computations by hand. She has a helper in form of a machine

(hard- or software), which we call Trent-A ("A" to denote that he works for Alice). Trent-A is a trusted entity and the keeper of Alice's signature. For the moment it is not important how the trust is established. For example, a government agency might vouch for him. Trent-A has sufficient computational power and signs Alice's documents in her name. Whether Trent-A is implemented as a smartcard attached to the PC, a tamper resistant program [1], a trusted computer in Alice's LAN, or similar, is irrelevant. The two work as a team: Alice, having written the document, passes it to Trent-A. He produces the signature and sends it back to Alice. Having now both the document and the corresponding digital signature, Alice sends them to Bob.

Notice that all communication between Alice and other actors goes through her PC. Unfortunately for Alice, there is another, malicious actor in this setup: Mallory, of whom Alice knows nothing. Mallory is an attacker, typically implemented as a virus or similar ubiquitous computer hazard, on Alice's PC. When Alice sends the document to Trent-A, Mallory modifies it on its way, or even foists her own. Trent-A, also not being aware of Mallory's presence, produces the signature for the altered document (the one he received) and sends the signature to Alice. Now Alice has her original document and a signature, but the signature corresponds to Mallory's document. Lacking computational power, Alice cannot notice Mallory's tampering — otherwise she would not need Trent-A at all and could produce the signatures herself. When she sends the document and the signature to Bob, Mallory again acts as man-in-the-middle, and exchanges Alice's document with the one produced by her. Bob thus receives a document which Alice never wrote, but which is properly signed in her name.

This makes the whole idea of digital signatures meaningless. Bob has, in principle, no more reasons to trust signed documents than unsigned. Digital signatures protect documents only once they are signed, but, in the above scenario, the attack happens before signing. Even if Bob trusts Alice's technical expertise and caution not to allow Mallory take over her computer, he still has a reason not to rely on such signatures: Alice, seeing that she cannot fulfill her part of the contract, might simply claim that she did not sign it, but that it was Mallory without her knowledge. Bob would be in trouble trying to prove otherwise.

The attack can also take place on Bob's PC in a similar manner: Mallory would sit in Bob's PC and intercept Alice's document as it arrives along with its valid signature. She would not interfere with the document or the signature as they are passed to Bob's trusted module (let's call it "Trent-B"), so the module would report that the signature is valid for the accompanying document. However, when Bob wants to see the document on his screen, Mallory foists her own. Bob has no way of noticing that and believes that the document he sees has been written and signed by Alice.[1]

III. Related technology

In both attacks, on the sender (Alice) and receiver (Bob) side, the problem arises from the fact that human users cannot be sure that the document they see on the screen is identical to the one received by the trusted module. There are basically three ways for solving it: equipping Trents with displaying capabilities, making the PC trustworthy, or securing the path between Alice and Trent-A and between Bob and Trent-B. The first two approaches are based on secure hardware, while the last one can be implemented in software.

A. External trusted device with displaying capabilities

If Trent (A or B) is an external trusted device, it is possible to provide him with an equally trusted display. Smartcard readers (terminals) with a LCD are common for electronic payment applications. Alice can trust the reader, e.g. because a trusted entity vouches for it. In the digital signature scheme, the display can be used for displaying the received document, so Alice could compare it to the document she sent. If the displayed document is identical to the original one, the signature can also be trusted.

The displays on smartcard terminals are usually small, low-resolution LCDs, large enough to display a price in the electronic payment scenario, but not large documents. Although it is technically possible to produce large, high-resolution secure displays, the economic viability of such devices, with the sole purpose of facilitating digital signatures, is questionable.

B. Trusted Computing

Economic viability could be achieved if the secure device is not specialized only for digital signatures, but of a general purpose. This is the approach pursued by the Trusted Computing Group (TCG) and supporting technologies, like LaGrande or TrustZone. The idea is to convert the whole PC, including the keyboard and the display, into a "trusted platform" (TP), while keeping it a general purpose computer, where the owner can install software or add peripheral devices. The security is guaranteed by the Trusted Platform Module (TPM), a cryptographic microcontroller built into the computer. Its purpose is to trace the changes to the computer hardware and software, so no changes — malicious or not — pass unnoticed by the owner. The trust in the platform is achieved through the chain of trust. At its beginning lies the first program that runs at power-on (e.g. the BIOS), which is required to be at least tamper-evident [9].

Leading manufacturers have initially announced TPs already for this year. Recently, however, it became clear that the full technology will not be a part of Microsoft's new operating

[1]This attack assumes that Trent-B can somehow bypass Mallory when reporting the result of signature checking. If Trent-B is implemented as an external hardware, e.g. a smartcard, the smartcard reader could light a green LED if the signature is valid for the document. If Trent-B has no way of bypassing Mallory, her attack is trivial: she can cut off Trent-B, impersonate him and claim for any document that its signature is valid (or not), as she wishes.

system, Vista, which is also behind the schedule. Customers also seem to be skeptical. It is feared, among other things, that through TC manufacturers might coerce users into using or not using some soft- or hardware, and that private information might be indirectly disclosed without user's knowledge [2]. That the fears are not baseless is indirectly confirmed by the TCG best-practice manual [13], which denounces such misuses of the TC technology, although they cannot be prevented.

IV. Authentication using visual cryptography and CAPTCHAs

A simple, low-tech alternative to the above approaches is visual authentication [7]. It is based on visual cryptography [8], a symmetric cryptographic method using visual secret key. In its basic implementation it is a visual implementation of the 2-out-of-2 secret sharing scheme [12]. The cyphertext is an image looking like a random distribution of black and white pixels. The secret key is a transparency, also with a random-looking pattern of black and white (actually transparent) pixels. Overlaying the key and the cyphertext, some areas will visually appear darker than others, and the "plaintext" will emerge. This "decryption" is performed with little conscious effort by the human visual system. Obviously, the method is only applicable for visually recognizable documents.

Symmetric cryptography cannot be directly used for ensuring message integrity. Mallory, who intercepts both the plaintext and the cyphertext between Alice and Trent-A, can easily compute the secret key and forge any message. In visual authentication scheme, Trent-A, before encrypting the document received from Alice, expands it with information unknown to Mallory, thus preventing her from deducing the secret. He sends the encrypted message to Alice, who visually decrypts it and checks if Trent-A received the document she wrote and if it is expanded as expected.

All expansion methods proposed in [7] — Content/Black Areas, Position on the Screen, and Black and Gray — require the secret key (the transparency) and the cyphertext to be about twice as large as the document. This is not an issue for electronic payment, which was the originally envisioned application, but might be impractical for larger documents, especially since each secret can be used only once, as a one-time pad. Instead, a it has been proposed for Trent-A to incorporate a faint image, a "watermark", into the document image before applying visual encryption. An example is shown in Figure 1.

Watermarks have been used in document security (especially counterfeiting) for centuries. Their use for digital documents, with different purposes in mind, has also been extensively studied (see, for example, [5]). In the document authentication scenario, the watermark which Trent applies has to be known to Alice (or, respectively, Bob), but unknown to Mallory — this can be achieved in practice by having a common trusted authority distribute the watermark to Trent and Alice (Bob) over a secure channel, e.g. digitally encrypted for Trent and per post for Alice and Bob. Also, to avoid replay

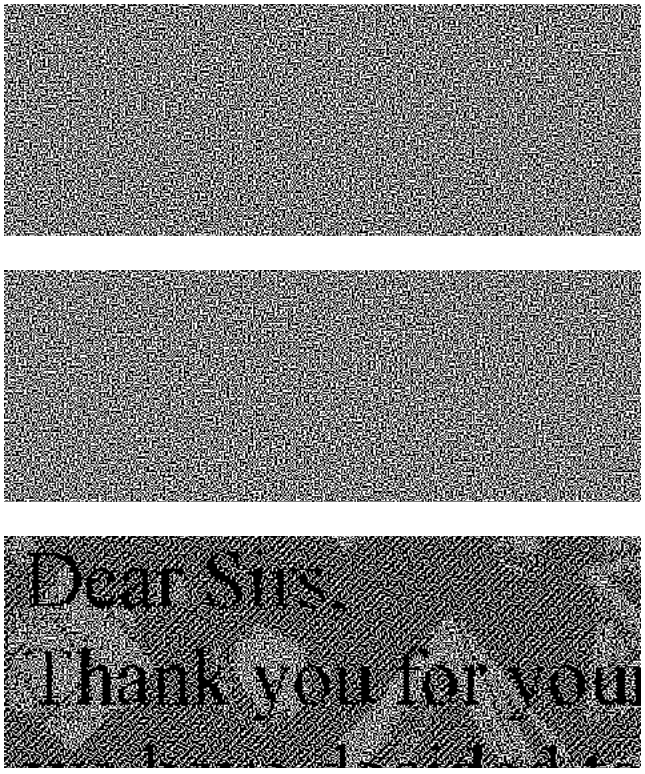

Fig. 1. An example of visual authentication: cyphertext (top), key (center), and decyphered message with the watermark (bottom).

attacks, each watermark should be used only once, so that a whole list should be distributed.

If Mallory tries to tamper with the watermarked and encrypted document, she will almost certainly either damage the watermark or produce an illegitimate share(see [4] for details). By visually checking the cyphertext-share and the integrity of the watermark, Alice and Bob can make sure that the document has not been tampered with.

Document authentication using visual cryptography has a practical drawback that it requires a separate instrument — the "secret key" transparency — which has to be carefully aligned with the cyphertext. For high-resolution document images this is a high-precision task, hard to achieve if the cyphertext is shown only on the screen, so printing it on the paper is the usual solution. This makes the technique slower, more expensive and thus somewhat less attractive.

An alternative is to use hard AI problems for document authentication. Exploiting the hardness of pattern recognition problems for security purposes has been proposed in [14]. It is generally known, not only in the pattern recognition community, that human capabilities to recognize objects are currently far ahead of computers'. Completely Automated Public Turing test to tell Computers and Humans Apart (CAPTCHA) relies on this property to distinguish between humans and computers. A CAPTCHA is basically a program that can generate and grade tests which most humans can pass, but not (current) computer programs. The idea is widely used on the Internet to block automated access to services, like e-mail accounts,

search engines, and similar.

An example of such test is Gimpy, an image containing several graphically distorted words. Humans can normally recognize some, or even all of them, but programming a computer to do so is everything but trivial. Obviously, the reliability of CAPTCHA degrades with the advancement of pattern recognition techniques. Gimpy can currently be solved by the computer in about 1/3 of the cases, and EZ-Gimpy (with only one word in the image) even in 92% [6], which is too much to make the test useful. Nevertheless, the range of still unsolved pattern recognition problems is wide enough to make CAPTCHAs attractive for security applications.

Using CAPTCHAs for document authentication has been proposed in [3]. The basic approach is similar to the one involving watermarks and visual encryption, but in contrast to the above scheme, a visual distortion of the document — a CAPTCHA — is used to hide information from Mallory. The process works roughly as follows: Alice writes the document and sends it to Trent-A for signing. On the way, Mallory can manipulate the document. Trent-A, having received the document (he does not know if it comes from Alice or from Mallory) distorts it, superimposes it with other visual information, and sends the result back to Alice. The visual distortion includes some faint, but visually recognizable image or pattern superimposed to the document, but can also involve deforming the document image itself. This is needed to prevent Mallory, who knows the plaintext, from simply "subtracting" it from the "encrypted" document and thus being able to more easily recognize the background pattern (watermark). So if Mallory intercepts the encrypted message on the way back to Alice, she cannot separate the document from the watermark, because she lacks the human pattern recognition capabilities. Thus, she cannot undo any changes she made to the document. Alice, being human, can easily recognize both the document and the watermark and check if the document is authentic.

CAPTCHA-protected documents need not be printed and can be displayed directly on the screen. This gives two additional degrees of freedom for constructing the watermark: color and time, making attacks even harder. An example snapshot is shown in Figure 2.

Fig. 2. A snapshot of an animated CAPTCHA for document authentication. The distorted text is projected onto a morphing surface. At the instant, the code "1234" can be recognized on the surface.

V. Cross-party document authentication

In the scenarios described in the previous section, visual encryption or employing a CAPTCHA is, in principle, sufficient to authenticate the document not only towards Alice, but also Bob. It is only required that Bob knows which watermark Trent-A is supposed to employ for authentication and, for visual cryptography, to have the same "secret key" transparency as Alice. Both can be shared between Alice and Bob like any other secret key, even in advance, since Alice also must know which watermark to expect from Trent-A. We can assume that the visual authentication method which prevents tampering with the document on Alice's PC is also secure enough to prevent manipulations on its way from Alice to Bob — this assumption holds unconditionally for visual cryptography, but should be taken with caution for CAPTCHAs, as we discuss below. Alice can send Bob the watermarked document, which he can directly display on his screen (or print in the case of visual cryptography). By checking that the watermark is as expected, he can be sure that the document is authentic.

In this setup, the biometrics of author's hand-written signature (image, hand-writing dynamic, pen pressure) could be used as a part of watermark, e.g. as an animation including the dynamic unique to the author[2]. Hand signatures have been accepted as an authentication method for centuries and would probably increase the acceptance of the proposed method. However, additional document transformations needed to thwart replay attacks still have to be investigated.

This approach has the advantage that only Alice needs a trusted module — Trent-B is not needed in the scenario. If the communication parties are symmetrical, i.e. equally likely to act as document senders or as receivers, this is of little relevance. Every document sending party has to be equipped by a trusted module, so if Bob wants to send Alice an authenticated answer, he will need his Trent anyway. On the other hand, much of the information flow is not symmetrical: think of television, radio, newspapers, WWW etc. A majority of the people consume more information than they produce. Also, in many cases — all of the above except the WWW — far fewer entities are involved in information production that in consumption. For example, a single TV station provides a program for a large audience. By extrapolation, it can also be

[2]We wish to thank the anonymous reviewer who brought this to our attention

assumed that, at least currently, more people are interested in receiving authenticated information — news, stock quotes, and similar — than in producing them. Since it is economically and logistically much easier to equip only a small number of publicly known entities with trusted modules than the whole population, this asymmetric approach can more easily be implemented in practice.

We might be tempted to use the asymmetric approach for bidirectional authenticated communication, by relying on a simple challenge-response protocol. Alice, with Trent's help, can send authenticated messages to Bob. If Bob wants to send an authenticated answer, he first sends it as a plaintext. Alice attaches a nonce to it, authenticates it, and sends it back to Bob. The nonce is not recognizable to Mallory, because it is visually encrypted or CAPTCHA-distorted in the authentication process. Bob checks if the received message is, apart from the nonce, the same as the one he sent and, if yes, sends Alice the nonce as the authentication code. Now Alice can be sure that the message she received from Bob is authentic. Notice, however, that the nonce is not a signature. Of the five characteristics of a signature (authenticity, non-reusability, non-alterability, non-repudiability and being unforgeable) [11], it has only the first two. Consequently, this simple protocol is of very limited use and breaks down if one of the parties is dishonest.

Even for one-way communication, or when both sides rely on a trusted module, authenticating messages between communication parties relying only on visual authentication has serious drawbacks. Recall that the watermarked document is actually an image or, in case of CAPTCHAs, maybe even an animation. In terms of memory consumption, such images tend to be at least 100 times bigger than original text documents, or more. For animated sequences, the inflation is even more drastic. It is hardly acceptable to send many megabytes through the Internet, only to authenticate a page or two of pure text. The other drawback is that watermarked documents, especially with animated CAPTCHAs, are almost impossible to convert back to a textual format — it is a *feature* of CAPTCHA-based authentication that the textual content of the document is not recognizable by computers. If Bob wants to further edit the document, to index it by keywords, or to use a similar text-based feature, his only option is to retype the document. This is, clearly, extremely user-unfriendly, so for any application where Bob might need the document in its original format, it will also have to be sent to him as such.

There is another reason to be cautious when using CAPTCHAs for end-to-end authentication. Their security is based on the fact that an attacker program running on user's PC cannot break them in given time, especially not without stealing all computing power. However, their security is hard to quantify: it is, basically, only an expert consensus, that specific pattern-recognition problem cannot be efficiently solved by computers. When CAPTCHAs are used for securing an asynchronous communication, like e-mail, the danger is that an attacker might employ many computers and invest much more time, thus drastically increasing his chances for reconstructing the watermark and forging the document.

Anticipating that eventually most people will want to be able to engage in bidirectional authenticated (and even confidential) communication and will thus have a trusted module on their disposition, we propose using a hybrid authentication method. In it, both visual authentication and classical cryptography are employed. The visual authentication is used locally, on Alice's and Bob's PC, for authenticating the communication between the human user and his or her trusted module (Trent). Classical cryptography is used for authenticating the communication between Alice's and Bob's computers. The workflow is as follows:

1) On the sender side, Alice passes her document to Trent-A and uses visual authentication to ensure that he received it unaltered.
2) If satisfied, she instructs him to digitally sign the document using ordinary (non-visual) cryptography and send her the signature.
3) She transmits the original text document and the corresponding signature to Bob.
4) Bob passes both the document and the signature to Trent-B
5) Trent-B cryptographically checks if the signature is valid for the document
6) If yes, he visually watermarks the document and sends it back to Bob. Essentially, Trent-B tells Bob: "*This* is the document which is signed by Alice" (actually it is signed by Trent-A, but Bob trusts him to work for Alice).
7) Bob compares the watermarked document with the plaintext. If their content is identical, he can be sure that the document comes from Alice and can use it as desired.

Beside the low memory consumption, this approach has the advantage that it does not substitute well-established digital security protocols, but complements them at points where they cannot be used. All the advantages of the classical cryptography are retained and additional security introduced at points where human and computer interface.

VI. Conclusion

We presented two protocols for authenticating digital textual communication between remote human users. The first method uses only visual authentication and is characterized by high bandwidth requirements. Its advantage is that only one communication party needs to have a secure, trusted module (hard- or software), which makes it appealing in setups where one large institution needs to send authenticated messages to many individual receivers. The second method is a hybrid protocol, where both classical cryptography and visual authentication are used. In communication this method does not require more bandwidth than classical cryptography, but requires all parties to use a trusted cryptographic module. This might be an obstacle for the acceptance of the method, since users might be reluctant to invest time and money in a new technology until it is widely spread. We estimate that the first method

currently has better chances of success. But, since the second one is technologically better, we expect it to succeed the first.

REFERENCES

[1] 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.

[2] 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

[3] 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.

[4] ——, "Visual document authentication using human-recognizable watermarks," in *Proceedings of ETRICS 2006, LNCS 3995*. Springer-Verlag, June 2006, pp. 509–521.

[5] F. Mintzer, G. Braudaway, and M. Yeung, "Effective and ineffective digital watermarks," in *Proceedings of IEEE ICIP 1997*, vol. 3, October 1997, pp. 9–12. [Online]. Available: citeseer.ist.psu.edu/mintzer97effective.html

[6] G. Mori and J. Malik, "Recognizing objects in adversarial clutter – breaking a visual captcha," in *Proc. Conf. Computer Vision and Pattern Recognition*, vol. 1. Madison, USA: IEEE Computer Society, June 2003, pp. 134–141. [Online]. Available: citeseer.ist.psu.edu/mori03recognizing.html

[7] 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.

[8] 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

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

[10] 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.

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

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

[13] 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

[14] 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