Using multiple smart cards for signing messages at malicious terminals

István Zsolt BERTA

Microsec Ltd.

This paper has been published at the
9th Information Security Conference August 30 - September 2, 2006, Samos, Greece, 2006.

Abstract. Having no trusted user interface, smart cards are unable to communicate with the user directly. Communication is possible with the aid of a terminal only, which leads to several security problems. For example, if the terminal is untrusted (which is a very typical scenario), it may perform a man-in-the middle attack. Thus, a malicious terminal can make the user sign documents that she would not sign otherwise. A signature that a card computes at a malicious terminal does not prove anything about the content of the signed document. What it does prove, is that the user did insert her card into a malicious terminal and she did intend to sign – something.

In this paper we propose a solution where a user has multiple smart cards, and each card represents a 'signal', a certain piece of information. The user encodes her message by using a subset of her cards for signing at the untrusted terminal. The recipient decodes the message by checking which cards were used. We also make use of time stamps from a trusted time stamping authority to allow cards to be used more than once.

1 Introduction

Electronic commerce applications require participants to send messages over a network. If these messages contain sensitive information, they need to be protected. For instance, a remote recipient who makes important decisions based on such a message would like to be convinced that the message is authentic: it originates from the sender and it has not been modified underway.

Digital signatures provide a way for a sender to ensure the authenticity of the message. Moreover, signatures allow a recipient to later prove it to a third party that the message originates from the sender. Today some signatures are even legally binding, so courts accept them as non-repudiable evidence.

In this paper that practical scenario is considered, where a human user would like to send a digitally signed message to a remote partner. Our user is mobile, and she does not have a trusted computer on her. All she has is a smart card that stores her private signing key. She would like to send a message of utmost importance, and she supposes that every terminal she can access is possibly malicious.

Signing messages at malicious terminals is dangerous. The digital signature is a very complex operation, the user is unlikely to be able to compute it without any computational aid. Typically, the signature is either computed by a terminal, or by the smart card of the user. It is unwise to trust a malicious terminal with computing a digital signature, because it may abuse the user's signing key. If the signature is computed by the smart card of the user, the terminal cannot get hold of the key itself. However, the smart card does not have any user interface of its own, so the user still has to rely on the malicious terminal for sending the message to the smart card before the card signs the message. In this step, the malicious terminal may perform an obvious attack: it may replace the message with another one that the user would not want to sign.

As a matter of fact, malicious terminal can make the user sign an arbitrary message.

2 Related Work

The problem of man-in-the-middle attacks of untrusted terminals was addressed by Abadi et al. first, by analyzing the dangers of delegation of rights to a terminal. [1] They show that this problem could be solved with a smart card that has peripherals to communicate directly with the user, and they also show secure protocols for such a device. Later on, they strip as much of these peripherals from the card, as possible. Schneier and Shostack also give a good overview of this problem. [2] Literature provide three branches of solutions for the problem of sending authentic messages from untrusted terminals. Some works (e.g. [3], [4], [5]) propose solutions based on *super smart cards*, that do have some peripherals for communicating with the user directly.

Some other works are based on *human-computer cryptography*, they provide solutions where the human user protects the message without the help of a smart card. Examples for this approach are visual cryptography, and the human authentication scheme of Matsumoto. [6], [7] These solutions rely on the fact the human can perform certain special operation much faster than computer can. There are some cryptographic algorithms that are optimized for being executed by humans. The “Solitaire” encryption algorithm is a good example for this. [8] We do not know of such an algorithm for message authentication.

In contrast to the above two branches, this paper provides a solution based on *realistic smart cards*, devices that exist today and that are feasible to deploy at large scale. In our solution the human user does not have to perform any cryptographic operations for authenticating the message.

In case of solutions based on realistic smart cards, it is assumed that the card does not have any peripherals for direct communication with the user. This means that the user, the card and the remote partner of the user are three entities that can communicate with each other only through the malicious terminal that is a part of the insecure channel. (See Figure 1.) Thus, establishing secure communication between the user and her card is exactly the same problem as establishing secure communication between the user and the remote partner.

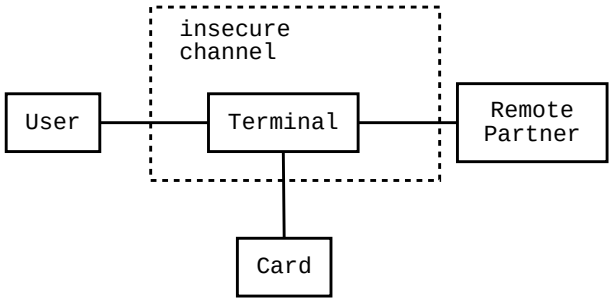

Fig. 1. A model for systems with malicious terminals

The model on Figure 1 was introduced by Stabell-Kulo et al. [9] who proposed that the user may authenticate her message by encrypting it using a one-time-pad and a monoalphabetic substitution table. Unfortunately, there are severe key management problems concerning the one-time-pad. For example, in case of long messages the user is unable to memorize the long one-time keys.

Asokan et al. proposed a solution that allows the user to authenticate untrusted terminals, but their solution does not deal with the problem of sending messages from malicious terminals. [10]

Berta and Vajda provided a formal model for the computational abilities a human user, and they have formally proven that this problem has no solution in case of messages that are longer than the key the user can memorize (which implies that one-time-pads cannot be used securely). They have proven that if the human user is unable to securely encrypt or authentiate a message without the help of a trusted computer, than the human is unable to take part in any protocol that would allow her to establish an encrypted or authenticated communications channel that the malicious terminal cannot easily attack. [11] This implies

that the problem of sending authentic messages from malicious terminals cannot be solved if we model the user as a slow computational network device. If there is a solution to this problem, it should be sought in a different model.

Berta et al. proposed a solution that change the model by posing lesser requirements on digital signatures computed at untrusted terminals. [12], [13] In this solution, the user can later revoke unintended signatures under well-defined circumstances. This solution is suitable for signing transactions of small value, but fails if the user can gain too much by repudiating a single signature. In contrast to the above work, signatures cannot be revoked in the solution we propose in this paper, so they can be used for authenticating transactions of arbitrary high value.

The solutions of Gruschka et al. ([14]) and Girard et al. ([15]) also pose lesser requirements on digital signature: in their model signatures cannot be used for any purpose. Their solutions protect the user by making the trusted smart card enforce limitations on what documents the user can sign. Thus if the card limits the messages that can be signed to small value bank transfers, the malicious terminal cannot make it compute a signature over a million-dollar contract. There are two problems with this approach: The first one is that this solution prevents the user from having a general purpose digital signature. The second one is that the malicious terminal can still alter the message within the limitations posed by the smart card. For example, if the card limits the messages that can be signed to million-dollar contracts, then the user would like to be absolutely sure, which million-dollar contract she signs. The work of Gruschka et al. also proposes solutions where certain parts of the terminal are secure, but in this paper we consider that particular situation only where the terminal is fully under the control of the attacker. The solution proposed in this paper does not place any restrictions on the messages the user can authenticate.

Berta and Vajda also proposed a solution [16] where the human user does not sign a plaintext message, but she signs a biometric (e.g. voice or video) message with her trusted smart card. The biometric message carries her biometric identity, so if the malicious terminal would like to make her sign a different message, then the malicious terminal needs to counterfeit the biometry of the user too. We assume that it takes significantly more time to counterfeit a biometric message than to counterfeit a plaintext one. There is a timestamping mechanism in the system that ensures that the malicious terminal has very little time to tamper with the biometric message before it is signed by the trusted smart card of the user. The solution we propose in this paper does not rely on the fuzziness of biometry, it employs purely algorithmic countermeasures.

3 Model

We can summarize previous solutions with the following statement: *A human user, who does not have a trusted computational device at her disposal, is helpless.* If the terminal she is using is malicious, it can make the user sign an arbitrary message. Previous works either assume that the user has some trusted computational device she can communicate with through a trusted channel, or they assume that the user has an exceptional memory or some exceptional computational abilities – which means that she (i.e. her brain) is the aforementioned trusted computational device. Some works try to protect the user by placing limitations on what documents she can sign, and prevent her from authenticating any message she chooses. Some others pose different requirements on signatures (and e.g. allow them to be revoked).

We do not know of any solution that allows the average human user to authenticate long messages without having a trusted computational device at her disposal that she can communicate with through a trusted channel.

In this paper, we propose such a solution. Our solution does not require the user to memorize any keybit or to perform any computations. Our solution includes trusted computational devices (smart cards), but does not assume that the user can communicate with them in a trusted way. What the user has to do, is allowing and blocking communication between the untrusted terminal and the smart cards. Naturally, the 'work' she has to perform is directly proportional to the length of her message. Our solution requires a global trusted time stamping service to be available.

According to the model we use in the rest of this paper, user U is a human who would like to send messages to remote partner, recipient R from malicious terminal T. The user has one or more smart cards that she can use for signing messages, and each smart card contains one signing key.

We define our model by the following assumptions:

1. The user, her smart cards and the remote partner can communicate with each other through the malicious terminal only (see Figure 1).
2. The terminal is fully under the control of the attacker, so the attacker is able to record, modify and replay any message passing through the untrusted terminal.
3. There is no cryptographic algorithm that a human user can execute to protect the authenticity of messages from the malicious terminal. This means that existing cryptographic algorithms are either too complex, so a human user cannot execute them on "long" messages (i.e. where one-time-pads are out of the question), or they are too weak, so a malicious terminal can easily break them. (See [11] for the formalization of this assumption.)
4. The user is able to block the communication channels between her smart cards and the malicious terminal. She has a straightforward way for doing this: the terminal is not able to communicate with a card unless the user inserts it into the card reader of the terminal.
   On the other hand, if the user inserts a card into card reader of the terminal, the terminal can make the card compute one or more[1] signatures over messages chosen by the malicious terminal until the card is removed from the reader. (We do not make any assumption on whether the card protects the signing keys by PIN codes. Although PIN codes are useful against e.g. card theft, they provide little protection against the threat of untrusted terminals, so their use is not discussed in this paper.)
5. The attacker cannot attack the digital signature algorithm with a non-negligible probability. This means that the attacker has a negligible chance to produce the signature for a given datablock, if the attacker does not know the corresponding private key. The attacker also has just a negligible chance for obtaining the signing key from a signature.
6. Smart cards generate their own signing keys, and it is impossible to extract these keys from the smart cards.

We assume that the attacker controls the terminal of the user when she sends messages m_1, m_2, ..., m_n. The aim of the attacker is to make the remote partner accept message m' as an authentic message originating from the user, where m' is not a message originating from the user ($\forall i : m' \neq m_i$). [17]

4 Our solution in a nutshell

According to Assumption 3, the user is unable to perform cryptographic operations that would allow her to send authentic messages to her smart card. This implies that the malicious terminal can make the user sign an arbitrary message, so in case of malicious terminals, *the digital signature of the card does not prove anything about the content of the signed message*.

However, a signature computed at a malicious terminal is not totally useless. Such a signature can still be used for proving the following: [16]

- The user signed *something* with her card.
- Whatever datablock was signed, it was not altered after the signing.
- If there are any securely obtained timestamps protecting the signature, they can be used for proving the time of the signing.

[1] It is possible to prevent the terminal from obtaining a signature without the user entering the PIN code by using trusted smart card readers with integrated PIN pads. However, not even a trusted reader with an integrated PIN pad can prevent the terminal from replacing the message the user would like to sign with an arbitrary message. In this paper we assume that the terminal is completely under the control of the attacker and it does not have any parts that are trusted by the user (Assumption 2).

In the solution we propose in this paper, we make heavy use of the first and the last of the above statements.

Let us assume that if the user wishes to send message ’1’ from a malicious terminal to the remote partner, and she inserts her card and signs a message. If the remote partner receives any message signed with the user’s private key, then the remote partner can make sure that the user sent message ’1’. If the remote partner does not receive any message signed by the user’s private key, then the remote partner cannot decide whether the user did not send anything, or she sent message ’1’ but the malicious terminal blocked it.

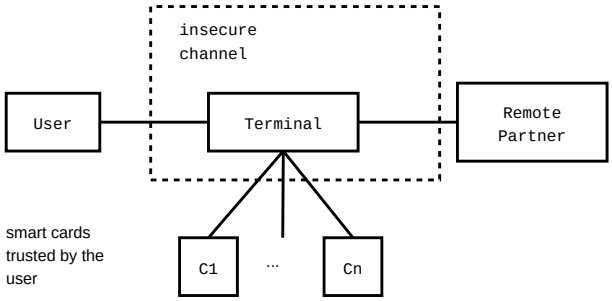

Fig. 2. Our model with multiple trusted smart cards

There are two problems with the above approach. On the one hand, the signing key (and the corresponding smart card) can be used only once. The user should not use the card ever again, otherwise the remote partner may not be able to differentiate between the received messages. On the other hand, it is not obvious how to send message ’0’.

We propose that the if the user has multiple smart cards, she can authenticate messages even at malicious terminals. In the solution we propose, the user does not authenticate the message by signing it with her card, but by signing some datablocks with subset of her cards. (Figure 2 shows the model with the user having multiple trusted smart cards.) Henceforth, we call a datablock and a signature on it created with one of the user’s cards a *signal*. Signals should have a structure that can be recognized by the remote partner, but the content of the signed datablock in the signal is irrelevant. Each card of the user contains a different private key, so each card produces a different signal. At the untrusted terminal, *the human user encodes her message as a list of signals*, and the remote recipient obtains the message by decoding the list of signals received from the user. As the malicious terminal is able to obtain signatures on any datablock, we assume that the digital certificates of the signing keys clearly state that signatures with the signing keys can be used for sending signals only, and such signatures do not prove the consent of the user.

There is a theoretically workable solution for sending an n-bit-long message from an untrusted terminal: The user should be equipped with $2n$ smart cards, cards $one[1..n]$ for sending signal ’1’, and cards $zero[0..n]$ for sending signal ’0’. If the user wishes to send message ’101’, she has to send three signals, i.e. she has to perform three signatures with three cards: one signature with card $one[1]$, another one with card $zero[2]$ and one with card $one[3]$. The remote partner expects three signals to arrive, and each signal to denote a bit at a different position in the message. Unfortunately, this solution allows the user to send only one message, and this message must have a fixed length. If n, the length of the message is large, then this solution requires the user to carry a truckload of cards on her – cards she cannot use ever again. We improve this solution in the next section by allowing the user to use cards more than once. More precisely, we allow her to send multiple messages of variable length while carrying a constant number of cards only.

5 A detailed practical solution

It is possible to reduce the number of cards required for sending the message by introducing the notion of time in our protocol. If the time of signing was included in the signals, the remote partner would be able

to detect if the malicious terminal tampered with the order of the signals before sending it to the remote partner.

We can safely assume that the user has a source of time independent from the terminal, e.g. a watch. (The biometric protocol of Berta and Vajda also uses this concept. [16]) According to Assumptions 1 and 3, the user cannot send the value of the exact time to the smart card in an authentic way. Unfortunately, most smart cards today do not have a timer of their own, so they need to rely on a trusted source of time, like a time stamping authority (TSA). A time stamping authority is a trusted party who puts digitally signed timestamps on incoming messages. A TSA who receives input x answers $timestamp(x) = (t_{current}, sign_{TSA}(t_{current}, x))$. The user trusts TSA for having a having a secure source of time and for functioning correctly. The TSA also handles its signing key in a secure way (i.e. Assumption 6 is true for the TSA).

We assume the user and the remote partner agreed on a set of time frames. The first frame is between t_1 and t_2, the second frame is between t_2 and t_3, etc. We assume that the time frames are of equal length, i.e. $\forall i,\ t_{i+1} - t_i = t_s$.

It should not be complicated to agree on these time frames and the time frames should not be kept secret. For example, the user and the remote partner may agree that a new time frame starts with every minute, every minute past 20 seconds and every minute past 40 seconds. In this example, we assumed that time frames are 20-seconds-long. The length of time frames may depend on how well the user's watch is synchronized with the TSA. In case of precise watches, the protocol is viable with shorter (e.g. 5-seconds-long) time frames too. (See the work of Sánta for description of a similar experiment. [18]) If the user's watch cannot be synchronized with the TSA, then longer time frames (of 30 seconds, 1 minute or even 10 minutes) can be used. Thus, there is a possible tradeoff between speed and time synchronization.

If user U would like to send the signal of card C to the remote partner R using the malicious terminal T, she needs to engage in the following protocol:

1. The user inserts card C into the reader.
2. $C \rightarrow T$: r /where r is a fresh random number generated by C/
3. $T \rightarrow TSA$: r
4. $TSA \rightarrow T$: $timestamp(r)$
5. $T \rightarrow C$: $timestamp(r)$
6. $C \rightarrow T$: $sign_C(timestamp(r))$ /if the timestamp is valid/
7. $T \rightarrow TSA$: $sign_C(timestamp(r))$
8. The user removes the card from the reader.
9. $TSA \rightarrow T$: $timestamp(sign_C(timestamp(r)))$
10. $T \rightarrow R$: r, $timestamp(r)$, $sign_C(timestamp(r))$, $timestamp(sign_C(timestamp(r)))$
11. R accepts the signal of the user if the digital signature of C is correct and both timestamps are within the same time frame (i.e. $\exists i$, where both time stamps are between t_i and t_{i+1}).

Note that most steps of the above protocol can be automated. In fact, the user needs to perform only two actions for sending a signal: she has to insert her card before she would like to send a signal, and she has to remove her card afterwards.

The user initiates the protocol by inserting card C into the card reader of the terminal. Steps 3 to 9 has to be performed in the same timeframe, otherwise the remote partner will reject the signal. After Step 9, the untrusted terminal obtains the signal. The signal itself is the message that the untrusted terminal sends to the remote partner in Step 10.

Using the above protocol the user may send various signals to the remote partner who can interpret a series of signals as a single message. There are two major principles the user and the remote partner must adhere to:

P1 **User**: After the user started sending a message she has to send a signal in every time frame until the end of the message. The user must not send more then one signal in a time frame, and she must not insert more than one card in a time frame into the card reader of the terminal.
Recipient: Every time frame in the message may contain one and only one signal. If it is not so, the remote partner should consider that the message has been tampered with.

P2 **User**: Apart from the signals used for transmitting messages, the user has to clearly mark the beginning and the end of the message. Otherwise the untrusted terminal could tamper with the message by chopping signals off from either the beginning or the end of the message. One possibility for marking the beginning and the end of the message is sending a special $startstop$ signal (using a dedicated smart card). The other possibility is to use a special combination of signals that may not appear during the message.
Recipient: If the remote partner receives a message that does not have a $startstop$ symbol at the beginning or it is not terminated by a $startstop$ symbol, then the remote partner should consider that the message has been tampered with.

In other words, messages should be started by and terminated by $startstop$ signals, and they may not contain any empty time frames. Each time frame in a message may contain one and only one signal.

The user is able to send binary messages of an arbitrary length if she has three smart cards: a card called one for sending message bit '1', a card called $zero$ for sending message bit '0', and a $startstop$ card for marking the beginning and the end of messages. See Figure 3 for an example.

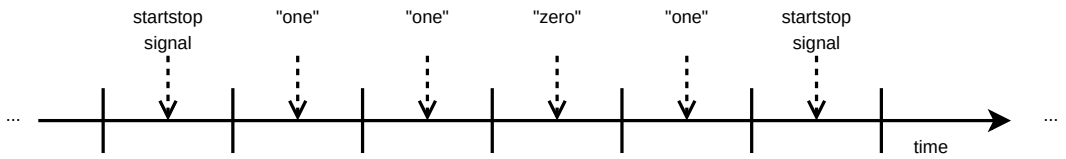

Fig. 3. An example

There is a possible tradeoff between the number of cards and the speed of protocol. The user and the remote partner may agree on a different set of signals, and on a different way of encoding the message into signals. For example, when sending numbers, the user may have one card for sending number '1', another for sending number '2', '3', etc. In case of sending text messages, it may be beneficial to select an encoding optimized for sending text.

In the next section, we are going to show that this protocol is secure against a certain attack model.

6 Analysis

We consider that the attacker controlling the terminal has possibilities described by the attack tree below. [19] The aim of the attacker is to make the remote recipient accept a message that the user did not sign as an authentic message originating from the user. (See Section 3 for the model of the attacker.)

1. The attacker may insert additional signals into a message.
2. The attacker may remove one or more signals from
   (a) the beginning of a message.
   (b) the end of a message.
   (c) the middle of a message (i.e. from parts other than the beginning and the end of the message).
3. The attacker may modify the message by changing one or more signals into different ones. The attacker can do this by removing a signal from the message and inserting a different one instead by

(a) obtaining a signature from a card that is used for sending a different signal, and obtaining the necessary timestamps from the TSA.
(b) forging a signature of a card that is used for sending a different signal, and obtaining the necessary timestamps from the TSA.
(c) obtaining a signature from a card that is used for sending a different signal, and forging the necessary timestamps of the TSA.
(d) forging a signature of a card that is used for sending a different signal, and forging the necessary timestamps of the TSA.

4. The attacker may perform cut-and-paste attacks.

Proposition 1. *If our assumption in Section 3 hold, then the solution we proposed in Section 5 is secure against the attacks in the above attack tree.*

Proof. We show that – according to our assumptions – none of the attacks on the leaf nodes of the attack tree are possible.

- Attack 1 is not possible. Between the starting and the terminating $startstop$ signals, every time frame in the message contains one and only one signal. If the attacker inserts an additional signal, then one of the time frames contain more than one signal (Principle P1 is violated), so the message is invalid and is rejected by the recipient.
- Attack 2a is not possible. If the attacker removes the first $startstop$ signal, then the message becomes invalid (because Principle P2 is violated), so it is rejected by the recipient.
- Attack 2b is not possible. If the attacker removes the terminating $startstop$ signal, then the message becomes invalid, because Principle P2 is violated. Invalid messages are rejected by the recipient.
- Attack 2c is not possible. If the attacker removes any one or $zero$ signals from the message, then an empty time frame appears in the message, so it becomes invalid because Principle P1 is violated. Invalid messages are rejected by the recipient.
- Attack 3a is not possible. According to Principle P1, the user does not allow the terminal to communicate with two cards in the same time frame. This means that the terminal is unable to obtain signature from a different card in the same timeframe.
  The terminal is able to obtain signature from a different card in a different time frame. Since both timestamps in a signal must reside in one time frame, the attacker cannot delay any step in the protocol for sending signals to make the valid signal appear in a different time frame.
- Attack 3b is not possible, because the attacker is unable to forge the signature of an unknown private key (Assumption 5) and the attacker cannot extract keys from smart cards (Assumption 6).
- Attack 3c is not possible, because the attacker is unable to forge the signature of an unknown private key (Assumption 5), and the attacker cannot extract the key from the TSA.
- Attack 3d is not possible, because neither Attack 3b nor Attack 3c is possible.
- Attack 4 is not possible, because each signal contains a timestamp that clearly marks time frame of the signal.

7 Conclusion

We propose that – using multiple smart cards – the user can sign or authenticate messages even at malicious terminals. The user can achieve this by blocking the communication channel between the terminal and certain cards and by allowing the terminal to communicate with other cards. It is not the datablocks she signs that contain the content of the message (because she has no means to guarantee the integrity of these datablocks before they are signed), but she encodes this content as a set of cards she uses for signing. In fact, the user signs her message by inserting smart cards into the reader of the terminal and by removing them.

We showed a protocol where she has a watch and she can access a trusted time stamping authority (TSA). Using this protocol, she can send messages of any length with a constant amount of cards. We have also proven that the protocol is secure against a certain attack model.

The solution we propose might sound awkward. However, we do not know of any other solution that allows the average human user to send long authentic messages without a trusted terminal.

References

1. Abadi, M., Burrows, M., Kaufman, C., Lampson, B.: Authentication and Delegation with Smart-cards. Theoretical Aspects of Computer Software: Proc. of the International Conference TACS'91, Springer, Berlin, Heidelberg (1992)
2. Schneier, B., Shostack, A.: Breaking up is Hard to do: Modelling security threats for smart cards. USENIX Workshop on Smart Card Technology, Chicago, Illinois, USA, http://www.counterpane.com/smart-card-threats.html (1999)
3. Balfanz, D., Felten, E.: Hand-Held Computers Can Be Better Smart Cards. Proceedings of USENIX Security '99 Washington, DC. (1999)
4. Gobioff, H., Smith, S., Tygar, J.D.: Smart Cards in Hostile Environments. In Proceedings of the 2nd USENIX Workshop on Electronic Commerce, Nov 1996, 23-28 (1996)
5. Clarke, D., Gassend, B., Kotwal, T., Burnside, M., Dijk, M.v., Devadas, S., Rivest, R.: The Untrusted Computer Problem and Camera-Based Authentication (2002)
6. Naor, M., Pinkas, B.: Visual Authentication and Identification. Lecture Notes in Computer Science, vol 1294 (1997)
7. Matsumoto, T.: Human-Computer cryptography: An attempt. In ACM Conference on Computer and Communications Security, pp 68-75 (1996)
8. Schneier, B.: The Solitaire Encryption Algorithm. http://www.counterpane.com/ solitaire.htm (1999)
9. Stabell-Kulo, T., Arild, R., Myrvang, P.: Providing Authentication to Messages Signed with a Smart Card in Hostile Environments. Usenix Workshop on Smart Card Technology, Chicago, Illinois, USA, May 10-11, 1999. (1999)
10. Asokan, N., Debar, H., Steiner, M., Waidner, M.: Authenticating Public Terminals. Computer Networks, 1999 (1999)
11. Berta, I.Z., Vajda, I.: Limitations of humans at malicious terminals. Tatra Mountains Mathematical Publications, vol. 29, pp 1-16 (2004)
12. Berta, I.Z., Buttyán, L., Vajda, I.: Privacy protecting protocols for revokable signatures. Cardis2004, Toulouse, France (2004)
13. Berta, I.Z., Buttyán, L., Vajda, I.: A framework for the revocation of unintended digital signatures initiated by malicious terminals. IEEE Transactions on Secure and Dependable Computing, vol. (Vol. 2, No. 3), pp. 268-272, July-September (2005)
14. Gruschka, N., Reuter, F., Luttenberger, N.: Checking and Signing XML Using Java Smart Cards. CARDIS 2004, Tolouse, France (2004)
15. Girard, P., Giraud, J., Gauteron, L.: Secure electronic signatures when Trojan Horses are lurking. e-smart 2004, Sophia Antipolis (2004)
16. Berta, I.Z., Vajda, I.: Documents from Malicious Terminals. SPIE Microtechnologies for the New Millenium 2003, Bioengineered and Bioinspired Systems, Spain (2003)
17. Maurer, U.: A Unified and Generalized Treatment of Authentication Theory. Proc. 13th Symp. on Theoretical Aspects of Computer Science (STACS'96), Lecture Notes in Computer Science, Berlin: Springer Verlag, vol 1046, pp. 387-398, 1996. (1996)
18. Sánta, r.: Smart Card based Application for Message Authentication in a Malicious Terminal Environment. Master's Thesis, Fachhochschule Darmstadt (2004)
19. Schneier, B.: Attack Trees. Dr. Dobb's Journal December 1999, http://www.schneier.com/paper-attacktrees-ddj-ft.html (1999)