hPIN/hTAN: A Lightweight and Low-Cost e-Banking Solution against Untrusted Computers*

Shujun Li[1], Ahmad-Reza Sadeghi[2,3], Sören Heisrath[3], Roland Schmitz[4] and Junaid Jameel Ahmad[1]

[1] University of Konstanz, Germany
[2] Darmstadt University of Technology and Fraunhofer SIT, Darmstadt, Germany
[3] Ruhr-University of Bochum, Germany
[4] Stuttgart Media University, Germany

Abstract. In this paper, we propose hPIN/hTAN, a low-cost hardware token based PIN/TAN system for protecting e-banking systems against the strong threat model where the adversary has *full* control over the user's computer. This threat model covers various kinds of attacks related to untrusted terminal computers, such as keyloggers, screen scrapers, session hijackers, Trojan horses and transaction generators.
The core of hPIN/hTAN is a secure and easy user-computer-token interface. The security is guaranteed by the user-computer-token interface and two underlying security protocols for user/server/transaction authentication. The hPIN/hTAN system is designed as an open framework so that the underlying authentication protocols can be easily reconfigured. To minimize the costs and maximize usability, we chose two security protocols dependent on simple cryptography (a cryptographic hash function). In contrast to other hardware-based solutions, hPIN/hTAN depends on neither a second trusted channel nor a secure keypad nor external trusted center. Our prototype implementation does not involve cryptography beyond a cryptographic hash function. The minimalistic design can also help increase security because more complicated systems tend to have more security holes. As an important feature, hPIN/hTAN exploits human users' active involvement in the whole process to compensate security weaknesses caused by careless human behavior.

1 Introduction

Nowadays e-banking becomes more and more popular all over the world. A 2010 survey of the American Bankers Association showed that e-banking is now the

* This paper was published in *Financial Cryptography and Data Security: 15th International Conference, FC 2011, Gros Islet, St. Lucia, February 28 - March 4, 2011, Revised Selected Papers*, *Lecture Notes in Computer Science*, vol. 7035, pp. 235-249, Springer, 2012. Companion web page (with a full edition of this paper): `http://www.hooklee.com/default.asp?t=hPIN/hTAN`.

most preferred banking method of bank customers [2]. There is no doubt that most users consider security as the most important issue about e-banking. The earliest and simplest defense protecting e-banking systems is user authentication based on static PINs. The end-to-end secure communications between the client and the e-banking server is typically achieved via the SSL/TLS protocol [14].

While SSL/TLS is considered secure, static PINs are prone to social engineering attacks, in which the users are spoofed to expose their PINs. One of the most prevailing social engineering attacks is phishing attack [17]. In its simplest form the attacker sends phishing emails to lure gullible users to disclose their PINs on a bogus e-banking web site. Once the attacker gets the PIN of a victim, he will be able to steal the victim's money by logging into the e-banking system. To provide higher security, two-factor authentication has been widely deployed by financial institutions for strengthening their e-banking systems. The most prominent two-factor authentication scheme used for e-banking is PIN/TAN, which uses static PINs for login and one-time TANs for online transactions.

While PIN/TAN can reduce the risk of simple social-engineering attacks like email based phishing, it does not offer any security against man-in-the-middle (MitM) attacks, in which the adversary controls the communication channel between the user and the e-banking server. In a typical MitM attack, the adversary establishes an SSL/TLS connection with the e-banking server and another one with the user, and then forwards the PIN and TANs from the user to the e-banking server as usual, but tampers with the transaction data in real time.

MitM attacks can be made stronger if the attacker partially/fully compromises the user's computer. This is possible due to the wide spread of malware over the Internet. Some malware can inject malicious code into the web browser, so that the attacker can do more than in MitM attacks: monitoring the user's input in the web browser, redirecting the user to a fake web site, modifying the contents of web pages shown in the web browser, and so forth. This kind of attacks are sometimes called man-in-the-browser (MitB) attacks [13]. Other malware such as Trojans or rootkits can even allow the attacker to take *full* control over the user's computer. In the worst case, all the software, hardware drivers, the operating system and even reprogrammable hardware are under the *full* control of the attacker, thus rendering the user's computer *totally* untrusted.

In this paper, we consider e-banking solutions against attacks related to *fully* untrusted computers, and call them "man-in-the-computer" (MitC) attacks. Depending on the contexts, MitC attacks have different names in the literature, e.g. malware-based attack or malicious software attack [33], Trojan attacks [25], content-manipulation attacks [21], transaction generator attack [16], and so on.

Since the main goal of MitC attacks is transactions manipulation rather than identity theft, it is clear that the corresponding solutions for secure e-banking aim at providing transaction authentication. Roughly speaking, there are two basic approaches to achieve transaction authentication: the first approach requires message authentication of the transaction data sent from the user to the server, and the second one requires secure transmission of the transaction data and a transaction-dependent TAN back to the user for re-confirmation of the requested

transaction. Normally, the first approach involves a trusted input method of the transaction data and a trusted channel for secure data transmission from the user to the server, and the second one involves a trusted out-of-band (OOB) or encrypted channel for data transmission from the server back to the user. The re-confirmation in the second approach is achieved by simply sending the transaction-dependent TAN back to the server without protection.

A typical solution in use is mTAN deployed by many financial institutions around the world [4, 23]. The mTAN system follows the second approach, and use the cellular network as the additional trusted OOB channel to send the transaction data and the transaction-dependent TANs back to the user via SMS. The user verifies the transaction data and then sends the TAN to the server to re-confirm the transaction. While mTAN is able to offer an acceptable level of security against MitC attacks, the OOB channel based on cellular network and a smart phone is not always available at the user side. Furthermore, the cellular network is not free from potential attacks [29]. In addition, the user's smart phone may also be infected by malware [28] and is still prone to some more advanced attacks such as SIM card swop frauds [24] and insider attacks from the telecommunication service providers [15]. The high complexity of today's smart phones may also lead to potential security holes induced by software bugs [22].

In addition to mTAN, there are many other e-banking solutions against MitC attacks. A lot of these solutions are based on hardware devices such as general-purpose personal computing devices (smart phones or PDAs), smart card readers or USB-tokens. Although many of them do work in practice, they all have non-trivial drawbacks, which include relatively high implementation/maintenance costs, dependence on an external network/server, low usability, doubtful security, and so forth. As far as we know, all hardware-based solutions depend on at least one of the following three components: second trusted channel, secure keypad, and encryption. Some solutions also require optical devices such as digital cameras or optical sensors.

Our contributions: In this paper, we propose hPIN/hTAN, the first (to the best of our knowledge) hardware-based solution against MitC attacks that depends on none of the following components: second trusted channel, secure keyboard, trusted third-party, encryption. The main design goal of the hPIN/hTAN system is to achieve a better tradeoff between security and usability with a low-cost and easy-to-use USB-token. The security requirements are guaranteed through a secure user-computer-token interface and two security protocols. The interface can also reduce or compensate careless errors made by humans who may become the weakest link in the whole system.

Paper organization: In the next section, we introduce our hPIN/hTAN system in detail. The security analysis of hPIN/hTAN is given in Sec. 3. The usability and deployment issues are discussed in Sec. 4. In Sec. 5, we overview related work, their drawbacks and compare hPIN/hTAN with existing solutions. The last section concludes the paper and gives some planned work in the future.

2 The Proposed hPIN/hTAN System

Our hPIN/hTAN system is composed of two parts – hPIN and hTAN, which protect the login process and online transactions, respectively. The hPIN part also protects the hTAN part from potential abuse by enabling it only after the user successfully passes the hTAN part. In the following, we discuss the model, notations, requirements and the two protocols involved, respectively.

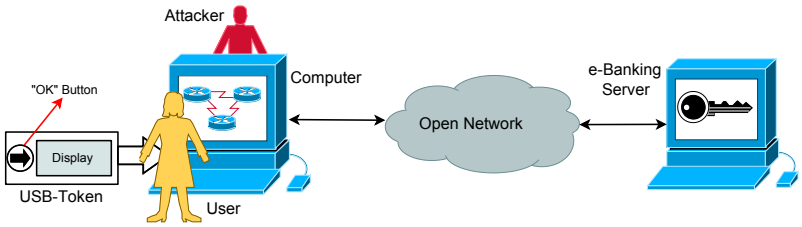

Fig. 1. The threat model of the hPIN/hTAN system.

System Model: As shown in Fig. 1, the involved parties in hPIN/hTAN are a human user U, a trusted USB-token T issued by the financial institute to the user, an untrusted terminal computer C (i.e., a MitC attacker), and the remote e-banking server S. In a typical scenario, the human user U plugs the USB-token T into a USB-port of the untrusted computer C, tries to access the remote e-banking server S and then makes some online transactions. We assume that the e-banking server S is trusted, which is a reasonable assumption in practice. The main threat we consider is the MitC attacker who is able to both observe and manipulate communications between U and C, T and C, S and C. Moreover, we assume the USB-token T is a trusted device to the user so that the MitC attacker has no access to any data stored inside T.

The notations used in this paper are summarized in the following table.

IDU	User ID.
K_T	Secret key shared between T and S.
PIN	n-character PIN shared between U and T.
PIN(i)	The i-th character of PIN.
s	Salt used to be hashed together with PIN.
STD	Sensitive transaction data that are authenticated.
NSTD	Non-sensitive transaction data that are not authenticated.
$h(\cdot)$	m-bit cryptographic hash function.
HMAC$(\cdot,\cdot)$	HMAC constructed based on $h(\cdot)$.
$a \parallel b$	Concatenation of a and b.
K_T^*	$= K_T \oplus h(\text{PIN} \parallel s)$ (stored in T).
PIN*	$=$ HMAC$(K_T, \text{PIN} \parallel s)$ (stored in T).
$\mathcal{F}_i : \mathbb{X} \to \mathbb{Y}$	Random code mapping PIN(i) $\in \mathbb{X}$ to a printable character in $\mathbb{Y}$.
C_T, C_S	Two counters stored in T and S.
V_T, V_S	Maximal numbers of consecutive failures allowed by T and S.

Security Requirements: Under the above system model, hPIN/hTAN is designed to achieve the following security requirements:

1. *PIN confidentiality*: the attacker cannot get the user's PIN in clear, except for a negligible probability;
2. *User authenticity*: the attacker cannot access the e-banking server without the presence of the legitimate user, except for a negligible probability;
3. *Server authenticity*: the attacker cannot cheat the user into connecting to a fake e-banking server, except for a negligible probability;
4. *Transaction integrity/authenticity*: the attacker cannot modify/forge a transaction without being detected, except for a negligible probability.

Note that the second and the third requirements are equal to mutual authentication between the user U and the server S.

System Requirements: The USB-token used in the hPIN/hTAN system is designed following a minimalistic principle. In addition to the basic components for building a USB device, it also includes a small display and an "OK" button. Two security protocols are embedded in the USB-token to implement user/server/transaction authentication. For our prototype system, we chose two security protocols based on an m-bit keyed hash function (HMAC). We avoid using any more cryptography to minimize the system complexity.

When a USB-token is manufactured, an m-bit secret key K_T and an initial PIN are assigned to it, where the PIN is composed of n characters in a finite set $\mathbb{X}$. The secret key K_T is crucial for the security of the hPIN/hTAN system, and is never shown in clear to the user and cannot be changed by the user. In contrast, the PIN is mainly used to protect the USB-token from theft and loss. As a whole, in the USB-token, the following data are stored:

$$\text{IDU},\ s,\ K_T^* = K_T \oplus h(\text{PIN} \parallel s),\ \text{PIN}^* = \text{HMAC}(K_T, \text{PIN} \parallel s),\ C_\mathsf{T},$$

where C_T is used to signal locking the USB-token if more than V_T wrong PINs have been entered consecutively. The salt s is used to frustrate rainbow table

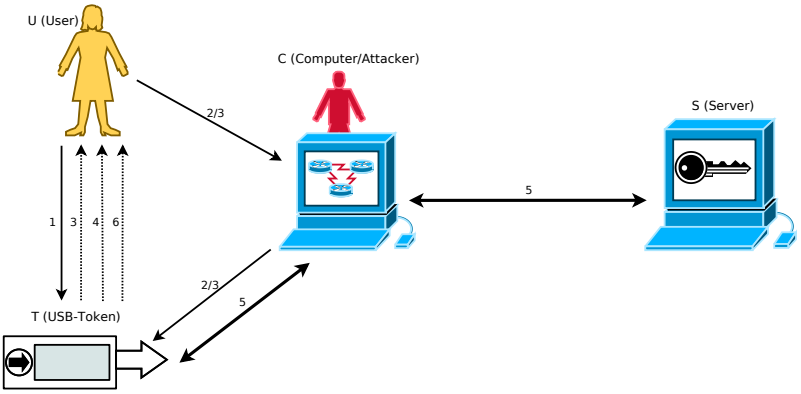

Fig. 2. The hPIN part, where solid lines denote real interactions/communications and dashed lines denote information display (the same hereinafter). The thick solid lines highlight the reconfigurable mutual authentication protocol.

attacks. Note that K_T is encrypted, and cannot be recovered without access to the correct PIN. The e-banking server stores the following data for the user:

$$\text{IDU}, h(K_T), C_{\mathsf{S}},$$

where C_{S} is used to signal locking the user's account if more than V_{S} consecutive failures of user authentication have happened.

Based on the above system requirements, the following two subsections describe how the hPIN and hTAN parts work. Note that running both parts needs installation of a plugin to the web browser of the terminal computer, which is in charge of communications between the USB-token T and the computer C.

2.1 The hPIN Part

The hPIN part protects the login process via the following two components: authentication of the user to the USB-token, and mutual authentication between the USB-token and the e-banking server. The second component can be implemented by any mutual authentication protocol. In this paper, we choose the SKID3 protocol [9], a generalized edition of the unilateral authentication protocol defined in ISO/IEC 9798-4. More complicated mutual authentication protocols can certainly be used here, but the simple SKID3 protocol is sufficient to achieve the security requirements of hPIN/hTAN. Thanks to the simplicity of SKID3, the computational complexity of the hPIN/hTAN is very low. Figure 2 and the following description explain how the whole hPIN part works.

Step 1: U connects T to C, and presses the "OK" button on T.
Step 2: U enters IDU on the untrusted keyboard and sends it to T via C.
Step 3: For $i = 1, \ldots, n$, T and U perform the following interactive protocol:

a) T randomly generates a one-time code $\mathcal{F}_i : \mathbb{X} \to \mathbb{Y}$, shows all codewords $\{\mathcal{F}_i(x)|x \in \mathbb{X}\}$ to U via its display;
b) U enters $\mathcal{F}_i(\text{PIN}(i))$ with the untrusted keyboard of C;
c) T decodes $\mathcal{F}_i(\text{PIN}(i))$ and performs $i = i + 1$.

Step 4: T verifies if $\text{PIN}^* = \text{HMAC}(K_T^* \oplus h(\text{PIN} \parallel s), \text{PIN} \parallel s)$. If so, then T recovers the secret key as $K_T = K_T^* \oplus h(\text{PIN} \parallel s)$, stores $h(K_T)$ in its volatile memory for future use, shows a "PIN correct" message to the user U via its display, and goes to Step 5; otherwise T performs $C_\mathsf{T} = C_\mathsf{T} + 1$, shows an alert to U and stops. If $C_\mathsf{T} > V_\mathsf{T}$, T locks itself.

Step 5: T and S authenticate each other by following a mutual authentication protocol. When the SKID3 protocol is used, the mutual authentication process works as follows:
$\mathsf{T} \to \mathsf{S}$: (UID,$r_\mathsf{T}$),
$\mathsf{S} \to \mathsf{T}$: ($r_\mathsf{S}$, $H_1 = \text{HMAC}(h(K_T), r_\mathsf{S} \parallel r_\mathsf{T} \parallel \mathsf{S})$),
$\mathsf{T} \to \mathsf{S}$: $H_2 = \text{HMAC}(h(K_T), r_\mathsf{T} \parallel r_\mathsf{S} \parallel \mathsf{T})$,
where r_S and r_T are nonces generated by S and T respectively.

Step 6: T shows a message on its display to inform U about the result of the mutual authentication protocol in Step 5.

After U successfully logs into the e-banking system with T, she can change the PIN if she wants. To do so, U asks C to signal T about the input of a new PIN. The new PIN can be entered in the same way as in Step 3 of the above hPIN process. After completing the PIN input, U presses the "OK" button on T twice and then T updates the values of K_T^* and PIN^*.

2.2 The hTAN Part

The hTAN part protects online transactions from MitC attacks after the user has successfully passes the hPIN process. As shown in the previous subsection, the hTAN part is enabled upon the completion of the hPIN process.

The core of the hTAN part is a human-computer-token interactive protocol that allows *simultaneous* transaction input on the untrusted keyboard of C and transaction verification via the trusted display of T. This interactive protocol ensures that T receives the real transaction data U wants to make. After that, T runs a transaction authentication protocol to send the real transaction data to S. In our prototype of hPIN/hTAN, we use the same HMAC scheme involved in the hPIN part for construct the transaction authentication protocol, so that the whole system is based on a single hash function and a single HMAC scheme.

Step 1: U clicks a button on the e-banking web page to inform T about the start of a new online transaction attempt. Then, she inputs each STD item one after another on the untrusted keyboard of C by repeating Steps 1–4.
To embed STD verification into the input process, each character in the STD is shown like passwords (e.g., as a number of asterisks) on the untrusted monitor of C, but in clear on the trusted display of T. This can naturally force U to verify the STD *simultaneously* while she is entering the STD.

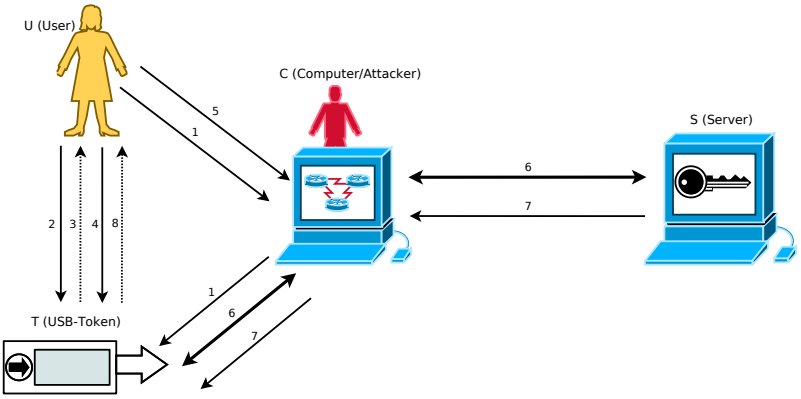

Fig. 3. The hTAN protocol. The thick solid lines highlight the reconfigurable transaction/message authentication protocol.

If U presses "Backspace" key, T shows an eye-catching warning message to inform the user for a few seconds and then the previously entered character is deleted. The goal of such a special design is explained later in Sec. 3.3.

Step 2: Upon completion of one STD item, U presses the "OK" button on T.

Step 3: T highlights the current STD item for a few seconds, and prompts U to press the "OK" button again.

Step 4: U presses the "OK" button again to approve the current STD item.

Step 5: U inputs NSTD to T by filling a blank on the web page in clear.

Step 6: T sends STD and NSTD to S by running a transaction/message authentication protocol. Here, we use HMAC to build the following protocol:
$\mathsf{T} \rightarrow \mathsf{S}$: (IDU, STD, NSTD, r_{T}^*),
$\mathsf{S} \rightarrow \mathsf{T}$: ($r_{\mathsf{S}}^*$, $H_3 = \mathrm{HMAC}(h(K_T), r_{\mathsf{S}}^* \parallel r_{\mathsf{T}}^* \parallel \mathrm{STD})$),
$\mathsf{T} \rightarrow \mathsf{S}$: (IDU, $H_4 = \mathrm{HMAC}(h(K_T), r_{\mathsf{T}}^* \parallel r_{\mathsf{S}}^* \parallel \mathrm{STD})$),
where r_{T}^* and r_{S}^* are two new nonces generated by T and S, respectively.

Step 7: S checks if $H_4 = \mathrm{HMAC}(h(K_T), r_{\mathsf{T}}^* \parallel r_{\mathsf{S}}^* \parallel \mathrm{STD})$. If so, S executes the requested transaction and sets $M =$"success", otherwise sets $M =$"error". Then, S sends $H_5 = \mathrm{HMAC}(h(K_T), r_{\mathsf{S}}^* \parallel r_{\mathsf{T}}^* \parallel M \parallel \mathrm{STD})$ to T.

Step 8: T checks if $H_5 = \mathrm{HMAC}(h(K_T), r_{\mathsf{S}}^* \parallel r_{\mathsf{T}}^* \parallel$ "success" $\parallel \mathrm{STD})$. If so, it shows "transaction executed", otherwise "transaction failed", on its display.

3 Security of hPIN/hTAN

In this section, we analyze the security of the hPIN/hTAN system, based on the assumption that $h(\cdot)$ and the HMAC scheme are both cryptographic secure against attackers whose computational power is limited by $2^{m/2}$.

3.1 PIN Confidentiality

The PIN protects the USB-token from theft and loss. Leaking the PIN to an attacker actually does not compromise the security of hPIN/hTAN (as long as

the USB-token is not available to the attacker), but it may compromise the user's privacy, since the PIN often relates to the user's personal information such as birthday. In addition, many users share the same PIN/password (or part of it) over multiple e-banking systems, so leaking the PIN of one e-banking system protected by hPIN/hTAN may lead to compromise of other e-banking systems.

The PIN confidentiality is achieved by the use of the n random codes $\mathcal{F}_1$, ..., $\mathcal{F}_n$ in the hPIN process. In Step 2, the USB-token T does not send the n codewords to the untrusted computer C, but shows them on its own display. Since the USB-token is a trusted device, the attacker has no access to any of the n codes and thus is unable to decode the PIN from the user's input $\mathcal{F}_1(\mathrm{PIN}(1)), \ldots, \mathcal{F}_n(\mathrm{PIN}(n))$. Each PIN character is mapped to a printable character by a *different* code, the attacker cannot figure out repeatedly used characters in the PIN, either. Instead, the attacker can only exhaustively search all the possible PINs. This corresponds to a success probability $|\mathbb{X}|^{-n} \ll 1$, when $|\mathbb{X}|^n \gg 1$. Note that an offline attack on the PIN is not possible because no information about the PIN is transmitted to C. An online attack is also impossible because Step 1 requires physical access to T. The above facts imply that the attacker has no any clue to judge if a random guess is correct or not, thus making the brute-force attack useless.

3.2 User/Server Authenticity

The mutual authentication between U (actually T) and S is guaranteed by the underlying security protocol in the hPIN part. For the SKID3 protocol, mutual authentication is guaranteed because the attacker can only passively forward communications between U (i.e., T) and S. That is, without the presence of U and T, the attacker cannot authenticate itself to S; without the presence of S, the attacker cannot authenticate itself to T. Note that we do not attempt to prevent the attacker from reading communications between U (i.e., T) and S, since they have been exposed to the attacker by the untrusted computer C.

3.3 Transaction Authenticity/Integrity

Transaction authenticity/integrity is achieved by the hTAN part. There are two stages of transaction authentication: 1) the human-token-computer interactive protocol in Steps 1–4 guarantees the integrity of STD from U to T; 2) the transaction/message authentication protocol in Step 6 guarantees the integrity of STD from T to S. Note that Step 8 is for the integrity of the "success" message from S, so it is independent of the integrity of STD and will not be discussed further.

The human-token-computer interactive protocol (Steps 1–4) ensures that T gets the STD without being manipulated. Since the user has to look at T's display to verify her input and then press the "OK" button twice to show confirmation, T will always receive the real STD that the user intends to input. Thanks to the use of the trusted display of T, the user can fully focus on the correctness of STD in the data input process. This is how *simultaneous* STD input and verification is achieved. The main goal of the special design on "Backspace" key is to prolong

the time of deleting previously entered characters so that malicious deletion of STD characters by C can be easily noticed by U.

Although Steps 2–4 look like "verify after input", the real purpose is to resist a competition attack: after U finishes typing the STD, the attacker sends one or more new digits to T and append them to U's STD. If this happens just before U's finger touches the "OK" button, U may wrongly confirm the attacker's STD. By asking U to press the "OK" button in Step 2 and then press it again after a short delay, the above competition attack will become nearly impossible. To detect an ongoing competition attack, U does not need to re-verify the whole STD explicitly, but just pay attention to possible abrupt change of the STD. This is a very easy task since U keeps watching T's display during Steps 1–4.

One may wonder why "simultaneous input and verify" is better than the traditional "verify after input" process. There are three main reasons: 1) recent research [1] has shown that human users are not dependable in distinguishing manipulated e-banking transactions (especially when only a few characters are manipulated) under the "verify after input" setting of mTAN; 2) we believe that asking the user to input and verify STD simultaneously can reduce the total time of STD input and verification (see Sec. 4 for more detail); 3) we believe that the user's active involvement at the very beginning of the hTAN process can help to enhance the user's feeling and awareness of e-banking security.

After T gets STD from the user, it sends STD to S for execution. The transaction authentication and re-confirmation is ensured by the two STD-dependent HMAC values H_3 and H_4. Under the assumption that the HMAC scheme is cryptographically secure, neither H_3 nor H_4 can be manipulated by the attacker with a non-negligible probability. Note that we also make the HMAC values depend on two new nonces to render replay attacks negligible.

4 Usability of hPIN/hTAN

We have developed a prototype implementation of hPIN/hTAN to test its usability. Three prototype USB-tokens have been produced and the hPIN/hTAN system has been implemented as firmware inside the USB-token and hostware running on a PC with Linux OS installed. Thanks to the simplistic design, the system is extremely lightweight: the size of the firmware is only around 10KB and the data memory requirement is less than 2KB. The actual costs of all components are about around 3–5 € per token. A virtual e-banking web site `http://www.hPIN-hTAN.net` was setup to simulate a genuine e-banking server. Figure 4 shows one prototype USB-token running Step 3 of the hPIN stage.

A small-scale user study with 20 students and staff members of our universities shown that with a 4-digit PIN the median login time is 27.5 seconds and a transaction with 55 characters can be completed in around 70 seconds (1.27 seconds per character). The overall success rate of logins is $60/66 \approx 91\%$. We expect the login time may reduce significantly after the user becomes more familiar with both the system and the PIN. A survey of the participants showed that none of them had major difficulties understanding how the system works. Most

Fig. 4. One prototype USB-token running the hPIN user authentication step.

users rated the overall usability of the hPIN/hTAN system as "very usable" and the mean opinion score is 3.65 on a 5-point scale. In the following, we give a qualitative analysis of the usability of hPIN/hTAN.

For the hPIN part, it is clear that the user interacts with T and C only in the first three steps and the following steps are done by T automatically. In Steps 1 and 2, the user only needs to press the "OK" button once and then input her ID, which does not add any additional usability problem compared with the traditional PIN scheme. The user interface in Step 3 is a bit more complicated due to the use of the random codes. To enhance usability, we propose to show the codewords of each random code $\mathcal{F}_i$ on T's display as follows (see also Fig. 4):

$$\begin{array}{cccccccc} 0 & 1 & \cdots & 8 & 9 & \cdots \\ \mathcal{F}_i(0) & \mathcal{F}_i(1) & \cdots & \mathcal{F}_i(8) & \mathcal{F}_i(9) & \cdots \end{array}$$

The first row lists all possible PIN characters and the second shows the corresponding code of each PIN character. This allows the user to simply look for her PIN character $\mathrm{PIN}(i)$ and then input the character below it. With a list of codewords as shown above, an average user can input each $\mathcal{F}_i(\mathrm{PIN}(i))$ within a few seconds. This means the whole PIN can be input within $O(n)$ seconds.

For the hTAN part, user interaction occurs only in Steps 1–5. Step 5 is about NSTD input, which is the same as the traditional TAN scheme, so we do not consider this step. The STD input in Step 1 is very similar to the normal text input in a web page. The only difference is that the user now should look at T's display rather than C's monitor to get visual feedback about what she is entering. By using a USB extension cable, the user can place T just above (or below) her keyboard so that she can easily see T's display. In this setup, the distance between the input device (C's keyboard) and T's display is much smaller than the distance between the input device and C's monitor, so the user is actually in a better condition of STD input. Steps 2–4 are very easy because the user either just waits and observes or simply presses a button on T. As a whole, we expect the additional time spent by an average user will be at the same order of traditional TAN schemes. Note that for TAN/PIN systems, the user has to look for the correct TAN on a paper list or wait for an SMS from the e-banking server, which can consume much more time than the hTAN process.

5 Related Work

As we mentioned in Sec. 1, transaction authentication is the key measure against MitC attacks, which can be achieved through two main approaches: 1) secure input and transmission of transaction data from U to S; 2) secure feedback from S to U for re-confirmation. In this section, we briefly overview previous solutions.

The first approach can be realized by transmitting the transaction data from U to S though an encrypted channel. For instance, in IBM ZTIC [33], a USB-token is used to establish a TLS channel for encrypting all communications between U and S. The USB-token has a trusted display and two buttons so that the user can explicitly confirm or cancel the transaction data. A low-tech solution called pTAN [7] is based on a paper list of secret permutation mappings, one of which is used for each transaction to conceal (encrypt) the input of transaction data from MitC attackers. Some other solutions are based on transaction-dependent TANs sent together with the transaction data to ensure transaction integrity. The TAN can be a MAC or a digital signature of the transaction data. A hardware device equipped with a secret key, such as a smart card reader [10, 31, 32] or a smart phone [26], is normally needed to calculate the TAN. To ensure that the TAN is calculated from correct transaction data, either a trusted keypad is necessary or the trusted hardware device reads the transaction data from the computer screen optically.

The second approach requires a trusted channel for U to receive the feedback from S. Some solutions use an out-of-band (OOB) channel like the cellular network [6, 23]. Other solutions use an encrypted channel. Different kinds of hardware devices are used for decrypting data sent from S, including smart phones [11, 19] and special-purpose devices like personal AXS-tokens [3]. Some solutions [3,11] also support direct readout of encrypted data from the computer screen. Visual cryptography and grille cipher are also used for this purpose [8,20].

Among all existing solutions, the simplest ones are based on CAPTCHAs [27, 30], which use CAPTCHAs as a virtual channel that cannot be handled by automated MitC attackers. However, [18] shows that almost all e-banking CAPTCHAs are insecure against automated MitM attacks. In addition, human-assisted attacks may always be used to circumvent e-banking CAPTCHAs [5,18].

Solutions based on low-tech "hardware" [7,8,20,27] have a major advantage: the implementation costs are relatively low compared with solutions based on electronic devices. However, these solutions often require the user to perform mental computations and/or align a paper sheet with the computer screen, thus reducing usability. In addition, low-tech "devices" are often less portable than small electronic devices. This problem becomes even worse when the user has to bring more than one such low-tech "device" [8, 27]. Furthermore, when a user wants to make a large number of online transactions in a short period of time, low-tech "devices" can be quickly used up, leading to a denial of service.

To save implementation costs, many solutions use smart phones and PDAs as trusted devices [6, 11, 19, 23, 26]. The most severe problem with such general-purpose devices is the potential risks of being infected by mobile malware [28]. Even worse, in order to circumvent the language or functionality limits set by the

manufacturers or the service providers, many users are willing to send their smart phones/PDAs to some private companies or alleged professionals to update the firmware, which makes it very easy for some attackers to inject malicious code into a large number of devices. In addition, as we point out for mTAN in Sec. 1, the high complexity of smart phones and PDAs leads to a higher risk of having security holes. If dependency on the cellular network is involved, then other weaknesses of mTAN will also be major problems.

In addition to mobile phones and PDAs, other trusted hardware devices used against MitC attacks include smart card readers [10,31,32], USB-tokens [33] and special-purpose devices like personal AXS-tokens [3]. All the smart card readers have a secure keypad as an essential component against MitC attacks. This not only increases the costs, but also reduces the device portability. In addition, some smart card readers are also improperly optimized to cause security flaws, and separation of the smart card and the reader leaves space for more potential risks [12]. The personal AXS-tokens do not have secure keypads, but are equipped with optical interfaces for reading data from the computer screen and biometric identification components, leading to a more expensive solution. Due to the need of maintaining an encrypted channel, devices without a secure keypad have an encryption module, which also increases implementation costs.

In comparison with other existing solutions, hPIN/hTAN is designed to reduce implementation costs without compromising security and usability. It uses a USB-token as the trusted hardware device, so it does not suffer from the problems with mobile phones and PDAs. Instead of using a keypad for secure input, we propose human-involved interactive protocols to create a trusted path from the untrusted keyboard of the untrusted computer to the trusted device. We also intentionally make hPIN/hTAN independent of an additional trusted channel and strong encryption. Such a minimalistic design not only leads to lower costs and better usability, but also to less software bugs and security holes. Table 1 shows a comparison of hPIN/hTAN and selected hardware-based solutions.

Table 1. Comparison of hPIN/hTAN with selected hardware-based solutions.

Solutions	*Smart phone/PDA*	*Secure keypad*	*Encryption*	*Data channel*	*External party*	*Smart card*
hPIN/hTAN	No	No	No	USB	No	No
mTAN [4, 23]	Yes	No	No	OOB	Yes	Yes
Sm@rtTAN plus [32]	No	Yes	No	No	No	Yes
Sm@rtTAN optic [31]	No	Yes	No	Optic	No	Yes
FINREAD [10]	No	Yes	Yes	USB	No	Yes
photoTAN [11]	Yes	Yes	Yes	Optic	No	No
QR-TAN [26]	Yes	Yes	Yes	Optic	No	No
IBM ZTIC [33]	No	No	Yes	USB	No	Optional
AXSionics [3]	No	No	Yes	Optic	Yes	No
MP-Auth [19]	Yes	Yes	Yes	Wireless	No	No

6 Conclusion

In this paper, we propose hPIN/hTAN, an enhanced PIN/TAN system based on a low-cost and easy-to-use USB-token, to protect e-banking systems from attacks related to untrusted computers, namely, man-in-the-computer (MitC) attacks. Our proposed system offers a better tradeoff between security and usability than existing solutions. The main feature of the system is the low complexity of the USB-token, which only needs to support a cryptographic hash function and some other simple functionalities. In addition, we carefully designed the protocols involved in the system to effectively exploit the human users' attention so that they will not be the weakest link in the system any more. Security analysis shows that hPIN/hTAN can achieve three security requirements: PIN confidentiality, user/server authenticity, and transaction authenticity/integrity.

We have developed a prototype system and performed a small-scale user study for demonstrating the usability of the hPIN/hTAN system. In the future we will investigate more variants of the basic design, and try to figure out if some variants have even better overall performance than the basic hPIN/hTAN system reported in this paper. For instance, we will study if the USB channel can be replaced by an optic or wireless one to enhance usability but with acceptable additional costs. We also plan to run further user studies to show the real performance of hPIN/hTAN for average bank customers.

Acknowledgments

Shujun Li and Junaid Jameel Ahmad were supported by the Zukunftskolleg of the University of Konstanz, Germany, as part of the "Excellence Initiative" Program of the German Research Foundation (DFG). Shujun Li and Roland Schmitz thank Prof. Walter Kriha of the Stuttgart Media University for valuable discussions and comments on an early draft of the paper. The authors also thank all participants of our user study running at the University of Konstanz and the Stuttgart Media University.

References

1. AlZomai, M., AlFayyadh, B., Jøsang, A., McCullagh, A.: An exprimental investigation of the usability of transaction authorization in online bank security systems. In: Proc. AISC'2008. pp. 65–73 (2008)
2. American Bankers Association: ABA survey shows more consumers prefer online banking. http://www.aba.com/Press+Room/093010PreferredBankingMethod.htm (2010)
3. AXSionics AG: Personal AXS-token (2009), http://www.axsionics.ch/tce/frame/main/414.htm
4. Bank Austria: mobileTAN information, http://www.bankaustria.at/de/19741.html
5. BBC News: PC stripper helps spam to spread (2007), http://news.bbc.co.uk/2/hi/technology/7067962.stm

6. Borchert, B.: Open sesame! – immediate access to online accounts via mobile camera phone, http://www2-fs.informatik.uni-tuebingen.de/~borchert/Troja/Open-Sesame/indexEN.php
7. Borchert, B.: Knick-und-Klick-TAN, oder Permutations-TAN (pTAN) (2009), http://www2-fs.informatik.uni-tuebingen.de/~borchert/Troja/pTAN
8. Borchert, B., Beschke, S.: Cardano-TAN, http://www2-fs.informatik.uni-tuebingen.de/studdipl/beschke
9. Bosselaers, A., Preneel, B.: SKID. In: Integrity Primitives for Secure Information Systems: Final Report of RACE Integrity Primitives Evaluation RIPE-RACE 1040, LNCS, vol. 1007, chap. 6, pp. 169–178. Springer (1995)
10. CEN (European Committee for Standardization): Financial transactional IC card reader (FINREAD). CEN Workshop Agreements (CWA) 14174 (2004)
11. Cronto Limited: Commerzbank and Cronto launch secure online banking with photoTAN – World's first deployment of visual transaction signing mobile solution (2008), http://www.cronto.com/download/Cronto_Commerzbank_photoTAN.pdf
12. Drimer, S., Murdoch, S.J., Anderson, R.: Optimised to fail: Card readers for online banking. In: Proc. FC'2009. LNCS, vol. 5628, pp. 184–200. Springer (2009)
13. Gühring, P.: Concepts against man-in-the-browser attacks (2007), http://www2.futureware.at/svn/sourcerer/CAcert/SecureClient.pdf
14. IETF: The Transport Layer Security (TLS) protocol: Version 1.2. RFC 5246 (2008)
15. IT-Online: World-first SMS banking scam exposes weaknesses (2009), http://www.it-online.co.za/content/view/1092105/142/
16. Jackson, C., Boneh, D., Mitchell, J.: Transaction generators: Root kits for web. In: Proc. HotSec'2007. pp. 1–4. USENIX (2007)
17. Jakobsson, M., Myers, S. (eds.): Phishing and Countermeasures: Understanding the Increasing Problem of Electronic Identity Theft. John Wiley & Sons, Inc. (2007)
18. Li, S., Shah, S.A.H., Khan, M.A.U., Khayam, S.A., Sadeghi, A.R., Schmitz, R.: Breaking e-banking CAPTCHAs. In: Proc. ACSAC'2010. pp. 171–180 (2010)
19. Mannan, M., van Oorschot, P.: Using a personal device to strengthen password authentication from an untrusted computer. In: Proc. FC'2007. LNCS, vol. 4886, pp. 88–103. Springer (2007)
20. Naor, M., Pinkas, B.: Visual authentication and identification. In: Advances in Cryptology – CRYPTO'97. LNCS, vol. 1294, pp. 322–336. Springer (1997)
21. Oppliger, R., Rytz, R., Holderegger, T.: Internet banking: Client-side attacks and protection mechanisms. Computer 42(6), 27–33 (2009)
22. PC World: Nokia: We don't know why criminals want our old phones (2009), http://www.pcworld.com/businesscenter/article/163515/nokia_we_dont_know_why_criminals_want_our_old_phones.html
23. Postbank: mTAN now free for all customers (2008), http://www.postbank.com/pbcom_ag_home/pbcom_pr_press/pbcom_pr_press_archives/pbcom_pr_press_archives_2008/pbcom_pr_pm1063_19_05_08.html
24. Saturday Star: Victim's SIM swop fraud nightmare (2008), http://www.iol.co.za/index.php?art_id=vn20080112083836189C511499
25. Schneier, B.: Two-factor authentication: Too little, too late. Comm. ACM 48(4), 136 (2005)
26. Starnberger, G., Froihofer, L., Goeschka, K.M.: QR-TAN: Secure mobile transaction authentication. In: Proc. ARES'2009. pp. 578–583. IEEE (2009)
27. Szydlowski, M., Kruegel, C., Kirda, E.: Secure input for web applications. In: Proc. ACSAC'2007. pp. 375–384. IEEE (2007)

28. The Financial Express: Russian phone virus that 'steals money' may spread global (2009), http://www.financialexpress.com/news/russian-phone-virus-that-steals-money-may-spread-global/420770
29. Toorani, M., Shirazi, A.A.B.: Solutions to the GSM security weaknesses. In: Proc. NGMAST'2008. pp. 576–581. IEEE (2008)
30. Volksbank Freiburg eG: iTANplus – mehr Sicherheit mit der indizierten TAN, http://www.volksbank-freiburg.de/itan.cfm?CFID=10869033&CFTOKEN=34249989&rand=1246061956151
31. Volksbank Rhein-Ruhr eG: Bankgeschäfte online abwickeln: Mit Sm@rtTAN optic bequem und sicher im Netz, http://www.voba-rhein-ruhr.de/privatkunden/ebank/SMTop.html
32. Volksbank Solling eG: Sm@rt-TAN-plus, http://www.volksbank-solling.de/flycms/de/html/913/-/Smart+TAN+plus.html
33. Weigold, T., Kramp, T., Hermann, R., Höring, F., Buhler, P., Baentsch, M.: The Zurich Trusted Information Channel – An efficient defence against man-in-the-middle and malicious software attacks. In: Proc. TRUST'2008. LNCS, vol. 4968, pp. 75–91. Springer (2008)