iPIN and mTAN for secure eID applications

Johannes Braun[1], Moritz Horsch[1], and Alex Wiesmaier[2]

[1] Technische Universität Darmstadt
Hochschulstraße 10, 64283 Darmstadt, Germany
{jbraun,horsch}@cdc.informatik.tu-darmstadt.de
[2] AGT Group (R&D) GmbH
Hilpertstraße 20a, 64295 Darmstadt, Germany
awiesmaier@agtgermany.com

Abstract. Recent attacks on the German identity card show that a compromised client computer allows for PIN compromise and man-in-the-middle attacks on eID cards. We present a selection of new solutions to that problem which do not require changes in the card specification. All presented solutions protect against PIN compromise attacks, some of them additionally against man-in-the-middle attacks.

Keywords: eID, iPIN, onetime PIN, nPA, mTAN, man-in-the-middle, PIN compromise, identity theft, smartcard

1 Introduction

1.1 Motivation

Electronic identity (eID) cards play an important role in trustworthy authentication and many countries already employ elaborated national eID cards. The German eID card [1], for example, provides machine readable travel document functionality as specified by the International Civil Aviation Organization [2,3,4] and is equipped with an eID functionality allowing the owner to electronically prove his identity. Furthermore, it supports an eSign functionality to generate (qualified) electronic signatures to be used in eBusiness and eGovernment applications. The eID and eSign functions are protected by separated personal identification numbers (PIN).

The German eID card is a representative of the newest generation of eID cards and may serve as blueprint for others cards to come. The card provides a contactless interface according to ISO14443 [5] and to supports version 2 of the Extended Access Control (EAC) protocol according to BSI-TR-03110 [6]. The EAC protocol provides a mutual authentication and may in particular be used together with the Password Authenticated Connection Establishment (PACE) [6, Section 4.2] protocol, which protects the communication over the wireless channel and ensures user consent. EAC and PACE have been proven secure against active adversaries having access to the communication channels between the involved components [7,8,9]. To use the card, a terminal is required where the user enters his PIN.

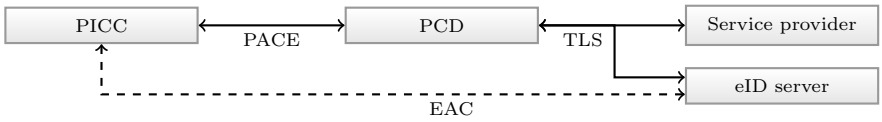

Fig. 1. eID infrastructure

However, terminals are not necessarily trustworthy, especially if the terminal consists of a simple card reader (without key pad) connected to a computer. Entering the PIN on a compromised terminal can leak it to an adversary and allow for identity theft attacks. Several attacks bypassing the security of the protocols based on compromised computers and eavesdropping on the PIN have been presented (cf. Section 1.3). The work at hand focuses on that threat concerning the eID functionality as implemented by the German eID card. We propose a new solution to protect the PIN without requiring changes to the card specification.

1.2 The eID functionality of the German identity card

The German eID card allows its cardholder to electronically prove his identity to service providers on the Internet. To use the eID functionality a terminal is needed, which in general consists of a computer connected to a card reader. A client application which implements the required communication and cryptographic protocols as well as the user interaction needs to be installed on the computer. In the following, we simply refer to the German eID card as the *card* or synonymously as Proximity Integrated Circuit Card (PICC) and to the terminal (including computer, card reader and client application) as Proximity Coupling Device (PCD). The terms PICC and PCD originate from BSI-TR-03110 [6]. As the legitimate user is always the cardholder we use the term *user* and denote his PIN with π to distinguish it from other constructs, such as temporary passwords.

In contrast to a common one-factor authentication by username and password, the eID functionality enables a two-factor authentication based on the ownership of the card and the knowledge of π. This enables a high level of trust between the service provider and the user which in fact is backed by a sovereign document. But it also implies big trouble if such a card is used illegitimately.

Any service provider (e.g. web mail service or online shop) that uses authentication via this card needs to present a certificate to the card to proof its identity and its permission for data access. These certificates are emitted by the German administration. However, the authentication process is not performed by the service providers themselves. Dedicated eID servers perform it in the name of the service providers. An eID server manages certificates issued for service providers, performs the security protocols, and reads the personal data stored on the card. The service providers only receive the data and perform a local authentication process based on their environment. The corresponding infrastructure is shown in Figure 1. The authentication process is as follows:

1. The user (using the PCD) opens the website of a service provider and clicks on a link to perform the login process and the client application starts.

2. The client application establishes a TLS connection to the eID server and receives the service provider's certificate.
3. The certificate description (e.g. issuer, URL, terms of usage) is displayed to the user, who agrees to the data transmission by entering the PIN π.
4. A secure channel is established between both parties by performing PACE.
5. The eID server and the card perform a mutual authentication using the EAC protocol. Hereby, the PCD serves as bridge between the secure messaging channel and the TLS channel. The certificate received in step 2 is used during the protocol to prove the access rights of the eID server.
6. The personal data is read by the eID server and passed to the service provider which grants access to the user upon receipt.

1.3 Identity theft attacks concerning the eID functionality

Given a non compromised PCD, the usage of the German identity card is secure. The necessity to enter a PIN to enable the communication with the card provides protection against the unauthorized usage of the card. Furthermore, the communication is secured against eavesdropping by the establishment of strong ephemeral session keys and encrypted communication. As detailed in this section, in case of a compromised PCD, the secure usage of the card cannot be guaranteed. As current attacks show, the major threat is PIN compromise.

PIN compromise attacks. As shown by the Chaos Computer Club [10] an adversary can obtain the PIN by using key loggers or Trojan horses. Once holding the PIN, the attacker still needs to access the card. Aside from stealing the card, the attacker can establish a remote connection to the card via the compromised PCD, if the card holder leaves the card on the PCD or is tricked into doing so. The adversary is then able to impersonate the victim.

Man-in-the-middle (MitM) attack. A MitM attack also requires the manipulation of the client application on the PCD. An adversary controlling the PCD presents the user the correct service certificate but sends another one to the card. Therewith, the user is tricked into entering his PIN and authenticating at a service different from the one he intended to. Taking over the session by the adversary after authentication and showing an error message to the user might leave the attack undetected.

1.4 Approach and outline

At present, the only way to prevent the attacks presented above is using card readers with a secure key pad and display. Such readers are expensive and might not always be available, for example in Internet cafés or other public places.

By adding an additional trustworthy identity provider, we propose a new solution working with a common basic card reader and leaving existing infrastructure components and protocols untouched.

In Section 2, we present the PACE protocol and address some background on multiparty computation. In Section 3, we present our solution involving different levels of trust. In Section 4, we give a security analysis and discuss the feasibility of our approach. The paper closes with future work and the conclusion.

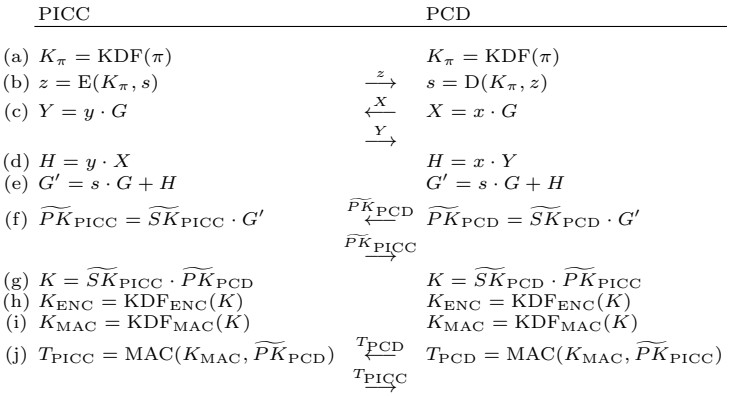

Fig. 2. PACE [6, Chapter 4.2]

2 Background

2.1 PACE

The Password Authenticated Connection Establishment (PACE) protocol [6] was developed by the German Federal Office for Information Security, is designed to be free of patents, and can be classified as a password-based key agreement protocol [11, Section 7]. PACE uses a password with low entropy to perform a user authentication and to establish a secure connection with strong ephemeral session keys. Usually, the password is a PIN π, which is permanently stored in PICC and is to be entered by the user into PCD for a successful protocol execution. Entering a wrong password leads to invalid session keys and the connection establishment fails. Generally speaking, PACE makes sure that only the owner has access to the card and unauthorized access is prohibited. PACE can be instantiated in different variants. Here, we focus on the elliptic curve variant with *Generic Mapping* [6, A.3.4.1] as used by the German eID card.

Figure 2 provides an overview of the protocol steps. Before it starts both parties agree on common domain parameters $\mathcal{D}$, containing elliptic curve parameters and a base point G. The numbers x, y, $\widetilde{SK}_{\mathrm{PCD}}$ and $\widetilde{SK}_{\mathrm{PICC}}$ are smaller than the order r of the elliptic curve and are chosen uniformly at random.

1. As depicted in step (a), both parties derive a key K_π from the shared password π using the key derivation function (KDF). The KDF enables to derive one or more secret keys from a common secret value and is basically a SHA-1 hash computation. In step (b) the PICC chooses a nonce s, encrypts it using the encryption function $\mathrm{E}(key, \cdot)$ with the key K_π, and sends the resulting ciphertext z to the PCD. The PCD decrypts z to obtain the nonce s.
2. In steps (c) – (e), both parties use s to generate a new common base point $G' = s \cdot G + H$, where H is agreed upon by the two communication partners in an anonymous Diffie-Hellman (DH) key agreement.

3. As shown in steps (f) – (g) a second DH key agreement based on G' is performed. Both, the PICC and the PCD choose an ephemeral private key ($\widetilde{SK}_{\mathrm{PICC}}$, $\widetilde{SK}_{\mathrm{PCD}}$) and calculate a common secret point K.
4. Steps (h) and (i) depict the key derivation of the keys K_{ENC} for encryption and K_{MAC} for message authentication from the common secret K.
5. In step (j) both parties calculate an authentication token (T_{PICC}, T_{PCD}) using a MAC function and the shared key K_{MAC}.

The authentication tokens (T_{PICC}, T_{PCD}) represent a mutual key confirmation and include a checksum of the ephemeral public keys ($\widetilde{PK}_{\mathrm{PICC}}$, $\widetilde{PK}_{\mathrm{PCD}}$). By checking the token, both parties can verify that the opponent calculated the same new base point G' and therefore knows the shared password π.

2.2 Multiparty computation

Perfect Secret Sharing means dividing a secret s into n so called shares, such that it is possible to reconstruct the secret if given at least $k \leq n$ shares. Less than k shares, however, provide absolute no information about the secret. I.e. the secret is information theoretically secure as long as an adversary only obtains less than k shares. An example is Shamir's secret sharing scheme [12].

Secure Multiparty Computation (SMPC) denotes the distributed computation of a function f by n participants. Thereby, a so called non qualified subset of $t < n$ participants cannot learn anything about the function output besides their own inputs and outputs. SMPC can be realized based on Shamir's secret sharing scheme, where $t < n/2$ is required [13,14,15] to guarantee perfect security against passive adaptive adversaries. Thus, $n = 3$ and $t = 1$ are the smallest possible parameters. Note that this means, that two participants can reconstruct the inputs and outputs without involving the third participant. An SMPC scheme providing computational security can be realized for $n = 2$ and $t = 1$ [16,14] based on the Paillier cryptosystem [17].

Secure Multiparty AES (MPC AES) is based on an SMPC scheme and implements the AES encryption and decryption interactively in a distributed manner as shown in [14]. For the MPC AES execution, each participant initially has to hold a share of the AES key as well as a share of the clear- or ciphertext. These have to be shared bytewise. At the end of the protocol execution each participant holds a share of the ciphertext (encryption) or of the cleartext (decryption). These shares can then be combined to a valid cipher- or cleartext.

3 iPIN and mTAN for eID cards

We provide solutions to prevent from the aforementioned identity theft attacks, even though the user only has access to an insecure PCD (e.g. without secure key pad and display). We use onetime passwords in different flavors, leading to the different solutions we present here.

To facilitate the use of onetime passwords and to ensure the desired security properties without changing the existing infrastructure and its protocols, we

introduce an additional trustworthy infrastructure component, called Universal Identity Provider (uIdP). Keeping existing components untouched is a crucial requirement, as several million German eID cards have already been issued and have to work with our solution. The uIdP is involved in the PACE protocol execution and interacts with an adapted client software installed on the PCD such that the involvement of the uIdP is transparent to the other components such as cards, eID servers, or service providers.

Concerning the onetime passwords, we differentiate between so-called iPIN and mTAN variants. iPIN denotes indexed onetime passwords, from which the PIN π can be reconstructed given the user's iPIN and the uIdP's iPIN with the same index. It is also possible to construct iPINs directly from K_π, which then can be reconstructed directly. From a security point of view, the knowledge of the derived key K_π is equivalent to the knowledge of π.

The mTAN variants are closely related to the mobile TransAction Number (mTAN) procedure known from online banking services. In online banking, a randomized TAN and some transaction details are send by the bank to the customer's mobile phone via SMS. The user confirms his consent by entering the TAN into the online banking application, thereby sending the TAN back to the banking server. The mobile phone is a so-called Out Of Band Device (OOBD). That means it realizes an additional communication channel (out of band channel) which is independent from the previously established communication channel between the two parties. In our mTAN variants, the uIdP takes the role of the banking server and sends the TAN to the user.

The several variants are justified by the different security goals they achieve, yet have their advantages and disadvantages. iPINs allow a stronger protection of the user PIN but require the precomputation and distribution of lists containing the iPINs, while the mTAN technique is more usable and allows to prevent from the MitM attack. We describe the variants along with their security assumptions, required setup steps, and detailed protocol steps in the following sections. Variants 1 and 2 apply iPINs while Variants 3 and 4 make use of the mTAN technique. Variant 5 combines both. Note that Variants 1 and 5 in addition to the uIdP technically require a second remote server we denote with uIdP-2. We refer the reader to Section 4.2 for a discussion on practical realizations.

Depending on the respective variant, a potential adversary requires different specific capabilities for a successful identity theft concerning different possible attacks. These capabilities are summarized in Table 1, which lists the attacks in its columns, the PACE variants (i.e. our solutions) in its rows, and the necessary attacker capabilities at the respective intersections. Basically, 'PIN comp. attack' and 'MitM attack' denote the attacks described in Section 1.3. Thereby, 'PIN comp. attack' only includes remote connections to the card, while 'physical card usage' denotes the physical theft of the card and its application to impersonate the owner. 'PIN compromise' means that the PIN is revealed to an adversary.

Regarding the attacker capabilities, `'r'` denotes read access, while `'rwx'` denotes read-write-execute access. Thus, e.g. `'uIdP:rwx'` means, that an adversary must be capable to read the uIdP's memory, as well as change and run

PACE variant	Threat: PIN compromise	PIN comp. attack	MitM attack	physical card usage
PACE	client:r	client:rwx	client:rwx	PIN + card theft
V1	two out of {client, uIdP, uIdP-2}:r	PIN + client:rwx	client:rwx	iPIN + card theft, OR PIN + card theft
V2	uIdP:r (during protocol run)	PIN + client:rwx	client:rwx	iPIN + card theft, OR PIN + card theft
V3	uIdP:r (anytime)	PIN + client:rwx, OR OOBD:r + client:rwx	client:rwx	OOBD + card theft, OR PIN + card theft
V4	uIdP:r (anytime)	PIN + client:rwx, OR OOBD:r + client:rwx	uIdP:rwx + client:rwx	OOBD + card theft, OR PIN + card theft
V5	two out of {client, uIdP, uIdP-2}:r	PIN + client:rwx	uIdP:rwx + client:rwx	OOBD + card theft, OR PIN + card theft

r = read access, rwx = read-write-execute access, theft = physical theft

Table 1. Required adversarial strength for identity theft

malicious code, whereas `'uIdP:r'` means only reading is required, which also includes key logging and so forth. With `'theft'` we denote the physical theft of things. `'PIN'` as an adversarial strength means, that the adversary must be able to compromise the PIN, implying the required strengths for that purpose.

3.1 Preliminaries

The preliminaries and principles are common to all five variants.

In general, we assume an adversary with the goal to compromise the PIN and/or perform the mentioned identity theft attacks but not to break up the flow of the protocols. Denial of service attacks are out of scope. Thus, even a compromised participant acts according to the protocol, as deviant behavior will be detected and responded with protocol abortion by honest participants, which is not in the interest of the attacker. The channels between PCD, uIdP, eID server, and service provider are always secured applying TLS, thus an adversary cannot eavesdrop on the communication between non compromised participants.

For each variant, a non recurring setup involving registration of the user at the uIdP(s) and establishing a unique user ID (e.g. a unique pseudonym) is necessary. The actual realization of the registration depends on the uIdP, but is essentially the same for all variants. In case of the involvement of two uIdPs, however, we assume for simplicity reasons that the user ID is the same for both. Depending on the actual variant, the setup phase involves the generation and transmission of pre-shared data such as iPINs to support later protocol executions. As this data is security sensitive, a non compromised system is necessary. Thus, we assume that this is temporarily available to the user during setup. We discuss how such a system can be practically provided in Section 4.2.

Protocol runs are initiated by the user. To do so, he starts the client application on the PCD, provides his ID, and applies his identity card upon request. As the uIdP needs the public domain parameters $\mathcal{D}$ (including the base point G) to enable dedicated computations of the PACE protocol, the PCD reads $\mathcal{D}$ from the card and initially sends it along with the user's ID to the uIdP.

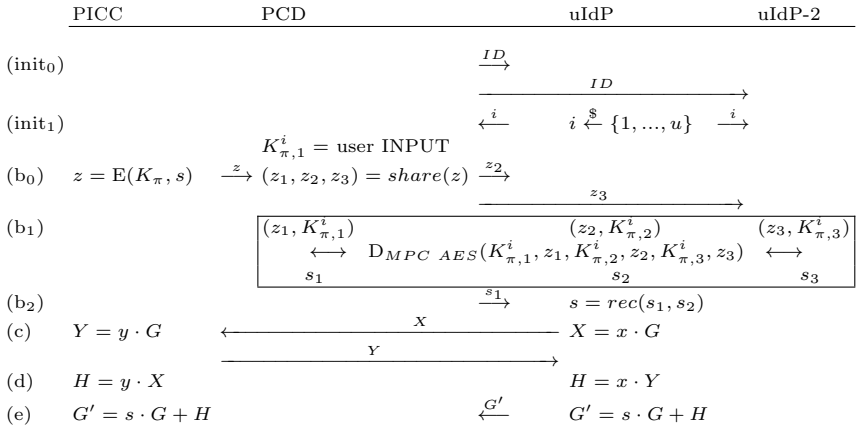

Fig. 3. PACE with multiparty decryption

3.2 Variant 1: Multiparty decryption of nonce

Variant 1 is designed to provide the highest possible security concerning the user PIN π respectively the derived key K_π. That is, they are never available in cleartext on any of the systems participating in the protocol execution. This goal is achieved by applying MPC AES to decrypt the nonce s. As MPC AES based on Shamir's secret sharing is necessarily a three party protocol [14], two trusted remote servers are required to implement this variant. We denote the trusted remote systems with uIdP and uIdP-2. The first uIdP server takes the main role, while the second one only provides support to facilitate MPC AES.

Assumptions. The adversary is able to compromise any participant, but at most one at the same time. uIdPs are trustworthy, meaning that they do not collude to obtain the PIN and act reasonably to protect themselves and the user.

Setup. To enable MPC AES, the key K_π must be shared in advance among the participants. The generation of iPIN lists works as follows: K_π is derived from π. Then, the sharing of K_π for three participants is repeated u times always storing each share in one of three indexed lists. Finally, each list has length u. Hence, they can be used u times before new iPIN lists must be generated. The lists are securely transmitted to the uIdPs and the user. E.g., the corresponding lists are sent to the uIdPs via TLS and the user receives a print-out.

Protocol execution. Figure 3 shows the steps of one protocol execution of Variant 1. To initialize the protocol run (step (init$_0$)), the PCD sends the user ID to the remote servers uIdP and uIdP-2. With the ID, the remote servers identify the user account and load the user specific information, e.g. iPIN lists.

To start the multiparty decryption, the shared key K_π must also be available to the participants. This was achieved by the distribution of the iPIN lists during the setup phase. To apply a specific iPIN, the uIdP chooses (uniformly at random) a fresh index i and announces it in step (init$_1$). Now, the user must

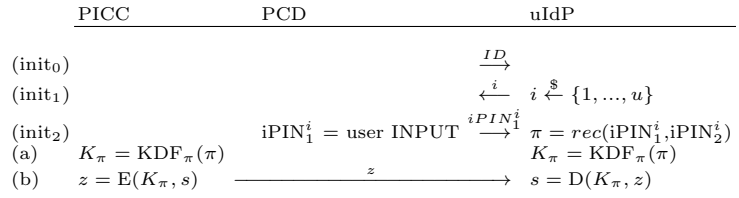

Fig. 4. PACE with PIN sharing

enter the correct iPIN $K^i_{\pi,1}$ taken from his list into the PCD to show his consent, while uIdP and uIdP-2 load the iPINs with the announced index from their lists.

Now the PICC gets involved. In step (b_0) the PCD receives the encrypted nonce z from the PICC. Remember, z is an AES ciphertext under the key K_π and does not reveal information about π. However, in combination with its decryption s, it is easily possible to brute force π. Thus, z must not be sent to the uIdP, to not expose both values to one system. The PCD shares z and sends one share to the uIdP, another one to the uIdP-2 and the third one is kept by the PCD.

In step (b_1) the PCD, uIdP and uIdP-2 jointly execute the decryption using MPC AES and each participant obtains a different share of the nonce s. The used iPINs are now securely deleted by uIdP and uIdP-2, thus a reuse of the user's iPIN (now known to the PCD) is impossible. From now on, the uIdP-2 is not needed anymore and it deletes all values computed during that session.

In step (b_2) the PCD transmits its share to the uIdP where s is reconstructed. The subsequent Diffie-Hellman key exchange (steps (c)-(d)) is executed between PICC and the uIdP. The PCD only forwards the data not being able to learn s.

In step (e) G' is transmitted from the uIdP to the PCD. Afterwards, the standard PACE steps are executed between PICC and PCD (see Figure 2).

3.3 Variant 2: Secret shared PIN

In Variant 2, the protocol is less complicated and much easier to implement than in Variant 1 as it requires only one trusted remote server and no MPC AES. However, the user PIN is temporarily revealed (by reconstruction from the iPINs) to the uIdP. But it can be deleted afterwards, thus limiting the time span the user PIN can be compromised by a potential intrusion into the uIdP.

Assumptions. Compared to Variant 1, we increase the security assumptions by requiring the uIdP not to be compromised during protocol execution. Also, the uIdP is trusted not to misuse the PIN, and to erase it securely after usage.

Setup. The generation and distribution of iPIN lists is as in Variant 1 with the differences that: first, π is shared instead of the derived key K_π and second, any perfect secret sharing scheme for two participants can be used, in particular XOR secret sharing which is very efficient and easy to implement.

Protocol execution. Figure 4 shows the protocol execution. Initially (step (init$_0$)), the PCD sends the user ID to the uIdP. With the ID, the uIdP again identifies the user and loads the specific information, in particular the iPIN list.

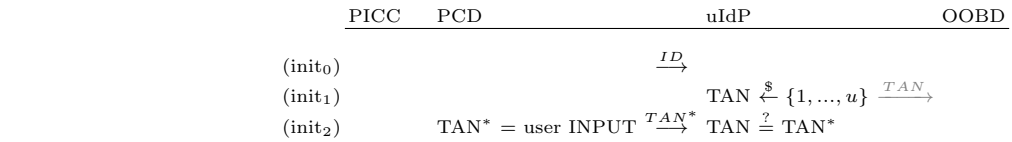

Fig. 5. PACE with mTAN

In step (init_1), the uIdP chooses (uniformly at random) a fresh index i and requests the ith iPIN from the PCD, thus from the user. In step (init_2), the user enters the correct iPIN into the PCD to show his consent. The iPIN is then sent to the uIdP, which reconstructs π and derives K_π (step (a)).

From now on, the PICC is involved into the protocol. z is sent by the PICC (see PACE specification [6]) and the PCD forwards it to the uIdP for decryption.

Afterwards, the protocol follows exactly Variant 1. After finishing its tasks, the uIdP deletes all session data including π, K_π and the used iPINs.

3.4 Variant 3: PACE with mTAN

This variant introduces the mTAN mechanism to circumvent the need for iPIN lists. The uIdP sends a onetime password (here called TAN) to the user's mobile phone, which serves as an OOBD. To show consent the user enters the TAN instead of an iPIN into the PCD upon request.

Compared to Variants 1 and 2, Variant 3 allows a much easier setup phase as no iPIN lists have to be generated and transferred in advance. Also, the usability is improved as the user is not required to keep a iPIN list (e.g. a piece of paper with iPINs printed on it). Yet the uIdP stores the user PIN π permanently leading to the disadvantage, that a compromise of the uIdP might reveal π at any time. Note that a large amount of PINs from different users might be revealed at once. Thus, advanced security mechanisms for the uIdP are indispensable.

Assumptions. In Variant 3, the security assumptions are further increased compared to Variants 1 and 2. The uIdP is assumed not to be compromised and is considered completely trustworthy.

Setup. Apart from the registration, only the user PIN π has to be transferred to the uIdP and the out of band channel must be defined. To define the out of band channel, the user might in particular deposit his mobile phone number at the uIdP, e.g. during registration or in a subsequent step, e.g. by yellow mail.

Protocol execution. As seen in Figure 5, the protocol execution of Variant 3 is very similar to that of Variant 2. The differences concern the initialization part of the protocol before the PICC is involved.

To initialize the protocol run (step (init_0)), the PCD sends the ID to the uIdP. With the ID, the uIdP identifies the account and loads π. Afterwards, the uIdP chooses a TAN uniformly at random, i.e. a six digit number and sends it to the user's mobile phone (step (init_1)). The uIdP appends additional information, e.g. to identify itself and the purpose and context of the message.

PICC	PCD	uIdP	OOBD
(init_0)		$\xrightarrow{ID,SI}$	
(init_1)		TAN $\xleftarrow{\$}$ $\{1,...,u\}$	$\xrightarrow{SI,TAN}$
(init_2)	TAN* = user INPUT	$\xrightarrow{TAN^*}$ TAN $\stackrel{?}{=}$ TAN*	

(a)-(j) *equal to standard PACE, though between PICC and uIdP, cf. Figure 2*

Fig. 6. Remote PACE

In step (init_2), upon receipt, the user enters the TAN into the PCD, to show his consent. The PCD forwards it to the uIdP, which compares it to the one it sent. If the TAN is positively verified, the uIdP proceeds, otherwise the protocol is aborted by the uIdP. Following a positive TAN verification, the PICC is involved and the following protocol steps are executed as in Variant 2.

3.5 Variant 4: Remote PACE and EAC

In Variant 4, the entire PACE protocol is executed remotely by the uIdP, while the PCD only forwards messages between card and uIdP. This means the uIdP is necessarily involved into the EAC protocol. Thus, the uIdP can check the service provider's certificate and reconfirm its identity by sending it together with a TAN to the user by applying the mTAN mechanism. Thus, this variant additionally provides protection from active adversaries and the MitM attack described above. On the negative side the user PIN π is permanently stored at the uIdP and the entire traffic of the eID functionality is routed over the uIdP.

Assumptions. The uIdP is assumed not to be compromised at all and is considered completely trustworthy. The PCD might try to manipulate the service's certificate as described for the MitM attack in Section 1.3. Thus, we assume a stronger adversary, that can manipulate the client application during execution.

Setup. The setup is identical to Variant 3 (cf. Section 3.4).

Protocol execution. Figure 6 shows the steps of Variant 4. Remember, that the service's certificate is obtained from the eID server before PACE is actually started and sent during EAC to the PICC after PACE was executed.

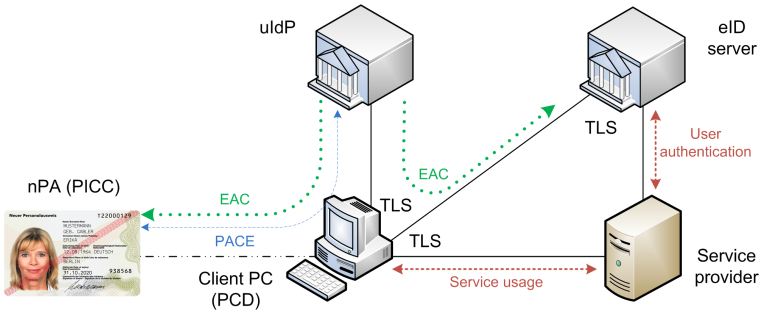

Fig. 7. Remote PACE and EAC Architecture

To initialize the protocol run (step (init_0)), the PCD sends the user ID and the service's certificate (SI) to the uIdP. With the ID, the uIdP identifies the user and loads the user's data. The uIdP chooses a TAN uniformly at random, i.e. a six digit number, and sends it to the user's OOBD (step (init_1)). Additionally, the uIdP appends the certificate's main information to the mTAN message.

That additional information allows the user to verify, that the certificate shown to him by the PCD and the one sent to the PICC are identical. In case this is true, the user enters the received TAN, which then is sent back to and verified by the uIdP (step (init_2)). A positive verification of the TAN, shows the user's consent. Then, the uIdP performs the standard PACE steps (a)-(j) jointly with the PICC. The PCD only forwards the data between uIdP and PICC.

As the complete PACE protocol is executed between PICC and uIdP, the PACE channel is established between them, and the PCD does not hold the keys for the PACE channel. Hence, all following messages for EAC must be routed over the uIdP for encryption when sent to and decryption when received from the PICC. The uIdP checks if the service's certificate is the same as the one the user confirmed during PACE. Only if this is true, the uIdP encrypts the certificate with the PACE keys and sends it via the PCD to the PICC. The PCD cannot tamper with that message as it lacks the keys.

Figure 7 shows the remote PACE and EAC architecture resulting from Variant 4. The PCD is the central element connecting the participants but mainly forwarding messages. The PACE channel is established between uIdP and PICC. During EAC, the eID server communicates via a TLS secured channel with the PCD that hands all messages to the uIdP to put these into the PACE channel and vice versa. Thus, from the point of view of the eID server the involvement of the uIdP is not visible. The same holds for the PICC.

3.6 Variant 5: Combination

Variant 5 is a combination of Variants 1 and 4. It makes use of iPINs and combines this with the mTAN mechanism and the remote PACE execution. Therewith, the reconfirmation of a specific service's certificate is possible. Hence, the user PIN π is never revealed and protection from MitM attacks can be guaranteed. However, this comes at the cost of the setup phase of Variant 1 and the uIdP being able to monitor the traffic of the eID functionality. As in Variant 1, a second uIdP-2 is required to support MPC AES decryption.

Note that other combinations of the different techniques used in the above presented variants are also possible, but not explained here.

Assumptions. We assume an adversary, that is able to compromise any participant, but at most one at the same time. Furthermore, the uIdPs are trustworthy in the sense that they do not collude to obtain the PIN and act reasonably to protect themselves and the user. This includes, for example, that the main uIdP is assumed not to store communication transcripts of the user.

Setup. The setup of Version 5 is the combination of the setups of Variants 1 and 4. It includes the generation of iPIN lists from the key K_π within a secure environment and their distribution (cf. Section 3.2). In addition, the user deposits

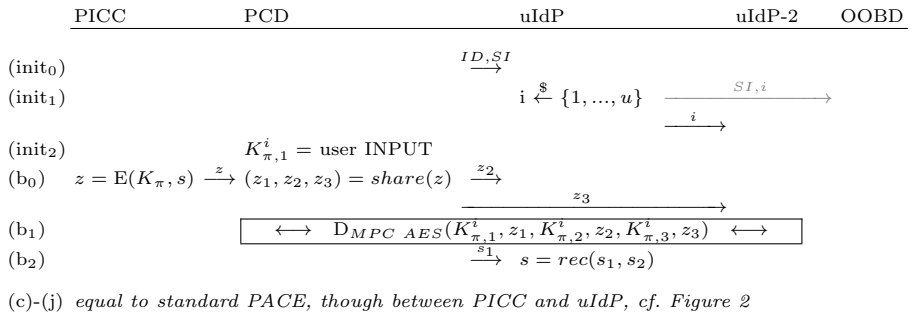

Fig. 8. Remote PACE with multiparty decryption

his mobile phone number at the main uIdP to enable out of band communication (cf. Section 3.5). Note that the PIN π is not transferred to any of the uIdPs.

Protocol execution. The protocol steps are depicted in Figure 8. To initialize the protocol run (step (init_0)), the PCD sends the user ID and the service's certificate (SI) to the uIdP. With the ID, the uIdP identifies the user and loads the user's data. In step (init_1), the uIdP randomly chooses an iPIN index i and sends i along with the service's certificate to the user's OOBD. At the same time the uIdP sends i to uIdP-2. In step (init_2), the user enters the correct iPIN $K^i_{\pi,1}$ taken from his list into the PCD to show his consent, while uIdP and uIdP-2 load the iPINs with the announced index from their lists.

In step (b_0), the PCD receives the encrypted nonce z from the PICC. The PCD shares z and sends one share to the uIdP and another one to the uIdP-2, while the third one is kept by the PCD. In step (b_1) the PCD, uIdP and uIdP-2 jointly execute the decryption function using MPC AES (see Section 2.2) and each participant obtains a different share of the nonce s. The used iPINs are now securely deleted by uIdP and uIdP-2. In step (b_2), the PCD transmits its share to the uIdP where s is reconstructed.

Then, the standard PACE steps (c)-(j) (see Figure 2) are executed between PICC and uIdP, while the PCD only forwards the messages between them.

4 Analysis

4.1 Security analysis

PIN compromise. The user PIN π or the derived key K_π can be compromised only if available on a compromised device or when it is computable from any of the exchanged messages or values computed during protocol execution. This is especially relevant, as the knowledge of a single cleartext-ciphertext pair of the nonce, i.e. s and z, allows for an offline brute force attack on the 6 digit PIN π.

The secret π cannot be compromised via the PCD in any of the variants, as: first it is never entered by the user. The entered iPINs in Variants 1,2 and 5 do not enable reconstruction due to perfect secret sharing as the other shares are never available on the PCD (uIdP and uIdP-2 do not reveal this values). The

TANs entered in Variants 3 and 4 are random numbers by definition and cannot reveal the PIN. Second, the PCD learns z, so the question is does it learn s? We show in the following, that it does not.

In Variants 1 and 5, z is decrypted to s by MPC AES. The multiparty computation based on Shamir's secret sharing scheme is unconditionally secure thus does not leak any information. The PCD does not get any of the other output shares besides its own one and s is reconstructed on the uIdP. In Variants 2 – 4, s is decrypted by the uIdP.

In Variants 1 – 3, the new base point G' is sent back to the PCD, ending the involvement of the uIdP(s). However, from G' one cannot recover s due to the difficulty of computing discrete logarithms. The steps after this mapping do not involve π, K_π, z or s besides their incorporation into G', hence need no further analysis. Note that in case of the remote Variants 4 and 5, the PCD actually has no informational advantage over a wiretapper eavesdropping on the contactless channel, and PACE has been proven secure against such an adversary [7].

In Variants 1 and 5, the PIN additionally cannot be compromised by intrusion into one of the uIdP's systems. The uIdPs learn their iPINs, one (unconditionally secure) share of z and the output of the secure multiparty computation respectively. The main uIdP additionally learns s from the reconstruction, but due to the lack of z cannot recover the user PIN.

In Variants 2 – 4, the PIN is revealed when the uIdP is compromised. The advantage of Version 2 is, that the PIN is only temporarily available, thus the compromise must occur at that certain time frame when the PIN is reconstructed.

To conclude, we note that a compromised participant, either PCD or uIdP, might reveal more than intended by the protocol, e.g. the PCD might reveal z to the uIdP. However, the other participant, not being compromised, would ignore such additional input or even detect the compromise and report it to the user.

MitM attack. This concerns only Variants 4 and 5, as the others do not provide protection from that attack.

An exchange of the service's certificate with another one by a compromised PCD is always revealed as explained in the following. If the uIdP receives the exchanged certificate during initialization, it sends it via the OOBD to the user who detects the exchange by comparison with the service's certificate. If the certificate is exchanged during EAC, the exchange is detected by the uIdP by comparison with the formerly received one. As the uIdP encrypts all messages sent to the PICC with the PACE key, which the PCD in both variants does not know, the PCD cannot exchange the certificate at any other protocol stage.

If the uIdP is compromised, the PCD is not compromised by assumption. The uIdP can now send an exchanged certificate to the PICC without being recognized by the user or PCD. However, the eID server applies the correct certificate as the PCD and eID server agree on the service without involving the uIdP. Thus, PICC and eID server apply different certificates and the mutual authentication fails.

We conclude that the MitM attack is prevented as long as either PCD or uIdP are not compromised.

Discussion. All variants hand over some control to the uIdP, which makes it an attractive target for attackers. But remember, any adversary additionally needs control over the eID card to maliciously use the eID functionality. This requires either additionally compromising the PCD or stealing the card. Furthermore, in case the user does not trust the uIdP anymore, he can change the PIN on his own at any time, thereby withdrawing all rights of the uIdP.

Besides that, the preliminaries for security are changed compared to the standard scenario. That is, the security in the standard scenario relies on two factors: the knowledge of the PIN and the possession of the card. This is changed to the possession of the card and the possession of either the iPIN list or the OOBD (e.g. mobile phone), which is an issue concerning physical theft. However, the factor 'knowledge' can be kept by requiring an additional user password when contacting the uIdP or accessing the OOBD. To provide the same level of security against physical theft, the password has to be handled in the same way as the PIN in the standard scenario. This means in particular that after three wrong entries the account is locked and a separate pre-defined substantially longer password is requested to unlock the account again.

4.2 Feasibility

For practical application several feasibility issues have to be discussed. One is the need for a trustworthy system accessible by the user for iPIN generation. This can be resolved by offering a secure system at the office of the authority that distributes the card. Another possibility is bundling the card with a bootable live CD containing a secure environment for iPIN generation. In this case, the user boots once in a while his own system from the live CD to create a new set of iPINs. It might even be reasonable to trust the user's home system with generating the iPINs. In this case, the security increase kicks in when using the card on other systems e.g. in Internet cafés. For the mTAN variants, which in fact do not require precomputations, the secure system could be omitted at all, e.g. by transferring his PIN, ID and phone number by yellow mail to the uIdP.

Concerning the required infrastructure, the need for two uIdP servers for Variants 1 and 5 is in question. Aiming at the highest possible security level, this comes with the registration at two independent service providers. Having one provider operating two independent servers requiring only one registration is more usable, but we have to trust the operator of the uIdP servers not to reconstruct the PIN. In both cases an adversary has to compromise two systems for a successful attack. Remember that 2-party SMPC is possible using Paillier's cryptosystem, but the application to our solution is left to future work.

The encryption with MPC AES takes two seconds [14] for one AES block based on a reference implementation [18]. As s has a length of 128 bit, only one AES block has to be decrypted. Hence, this is not a serious performance issue. Yet, the implementation of the multiparty computation is clearly non standard. The other variants only involve standard methods, as mTAN used for online banking and elliptic curve cryptography provided by several major crypto providers such as Bouncy Castle [19].

In practice, the uIdP service(s) could be provided by governmental authorities as part of the eID card infrastructure which is necessary anyway. It is also possible to have the private economy provide the uIdP services, as done with PKI services. Besides certificate authorities, banks seem to be reasonable candidates. The trusted infrastructure is already available as well as methods such as mTAN. The bank could also provide the system for iPIN generation at its offices.

Except the client application, all components of the standard infrastructure remain unchanged. The client application hides the involvement of the uIdP from the eID server, the service provider, and the PICC. Thus, they adhere to the standard protocols. Long delays might indeed lead to an abortion. Yet, with current high speed Internet connections we do not consider this to be a problem.

5 Conclusion and future work

We have shown five PACE variants that increase the security against identity theft. Our approaches allow the secure usage of identity cards, even though no trusted system is available to the user. All variants prevent from PIN compromise in case of a compromised client. Variants 4 and 5 even provide protection against the described MitM attack. The only security requirement is a trustworthy identity provider. It is a valid assumption that, as a part of its core business, the uIdP's security mechanisms are far more sophisticated than the ones on a usual client PC. The necessity of iPIN lists might lead to a decrease of usability. But for a scenario, where the user applies the card for authentication mainly from his home, this seems to be an acceptable effort compared to the increase in security. For the mobile scenario, where the user applies his card en route, the mTAN based approaches allow a secure and convenient usage even from Internet cafés. Thus, we provided possibilities to securely apply smartcard based authentication using insecure devices only, by adding a special infrastructure component to allow one time passwords without requiring any changes in existing protocol implementations on the smartcard or the existing infrastructure.

As mentioned in Section 2.2, using the Paillier cryptosystem Variants 1 and 5 can also be implemented as two party protocols with the advantage, that only one remote server is needed which can still not learn the PIN. But there are several drawbacks: first, the setup is much more complicated, in particular the iPIN generation. Second, MPC is only computationally secure, not being prohibitive but clearly needs further consideration concerning the security parameters to not weaken the overall protocol. Third, there exist no timings and performance estimations. Hence, the Paillier-based solutions are left for future work.

One of the next tasks will be the implementation of the introduced PACE variants based on an existing client implementation such as MONA [20]. The remote execution of the PACE protocol (Variant 4) has additional applications in scenarios where the card is used with resource restricted PCDs, such as a mobile phone as considered in [21,22,23,24]. Variant 4 hands all expensive computations to the remote server and the PCD only forwards messages. Thus, the client software on the PCD is much less involved and easy to implement.

References

1. Federal Office for Information Security. Architektur elektronischer Personalausweis und elektronischer Aufenthaltstitel. Technical Guideline BSI-TR-03127, Version 1.14, 2011. https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR03127/BSI-TR-03127_pdf.pdf.
2. International Civil Aviation Organization (ICAO). Machine Readable Travel Documents - Part 1: Machine Readable Passport, Specifications for electronically enabled passports with biometric identification capabilities. ICAO Doc 9303, 2006.
3. International Civil Aviation Organization (ICAO). Machine Readable Travel Documents - Part 3: Machine Readable Official Travel Documents, Specifications for electronically enabled official travel documents with biometric identification capabilities. ICAO Doc 9303, 2008.
4. International Civil Aviation Organization (ICAO). Supplemental Access Control for Machine Readable Travel Documents. ISO/IEC JTC1 SC17 WG3/TF5 for ICAO, Version 0.8, Draft of 12.10.2009, 2009.
5. ISO/IEC. ISO/IEC 14443-1: Identification cards - Contactless integrated circuit(s) cards - Proximity cards - Part 1-4. International Standard, 2001.
6. Federal Office for Information Security (Bundesamt für Sicherheit in der Informationstechnik). Advanced Security Mechanism for Machine Readable Travel Documents - Extended Access Control (EAC), Password Authenticated Connection Establishment (PACE), and Restricted Identification (RI). Technical Directive (BSI-TR-03110), Version 2.05, 2010. https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TR03110/TR-03110_v205_pdf.pdf.
7. Jens Bender, Marc Fischlin, and Dennis Kügler. Security Analysis of the PACE Key-Agreement Protocol. In *Information Security Conference*, volume 5735 of *LNCS*. Springer, September 2009.
8. Markus Ullmann, Dennis Kügler, Heike Neumann, Sebastian Stappert, and Matthias Vögeler. Password Authenticated Key Agreement for Contactless Smart Cards. *Communications of the ACM*, 2008.
9. Özgür Dagdelen and Marc Fischlin. Security Analysis of the Extended Access Control Protocol for Machine Readable Travel Documents. In *13th Information Security Conference*, Lecture Notes in Computer Science. Springer, October 2010.
10. Chaos Computer Club. Practical demonstration of serious security issues concerning swissid and the german electronic identity card., 2010. online: November 01, 2010. http://www.ccc.de/de/updates/2010/sicherheitsprobleme-bei-suisseid-und-epa.
11. Colin Boyd and Anish Mathuria. *Protocols for Authentication and Key Establishment*. Springer, 2003.
12. Adi Shamir. How to share a secret. *Communications of the ACM*, 22:612–613, 1979.
13. Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, STOC '88, pages 1–10, New York, NY, USA, 1988. ACM.
14. Ivan Damgård and Marcel Keller. Secure Multiparty AES. In *Financial Cryptography*, pages 367–374, 2010.
15. Ronald Cramer, Ivan Damgård, and Ueli Maurer. General secure multi-party computation from any linear secret-sharing scheme. In *Proceedings of the 19th*

international conference on Theory and application of cryptographic techniques, EUROCRYPT'00, pages 316–334, Berlin, Heidelberg, 2000. Springer-Verlag.
16. Ronald Cramer, Ivan Damgård, and Jesper Buus Nielsen. Multiparty computation from threshold homomorphic encryption. In *Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques: Advances in Cryptology*, EUROCRYPT '01, pages 280–299, London, UK, 2001. Springer-Verlag.
17. Pascal Paillier. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In Jacques Stern, editor, *Advances in Cryptology– EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238. Springer Berlin / Heidelberg, 1999.
18. VIFF. VIFF, the Virtual Ideal Functionality Framework, 2012. online: Jan. 19, 2012. `http://viff.dk/`.
19. Bouncy Castle. Bouncy Castle Crypto APIs, 2012. online: Jan. 19, 2012. `http://www.bouncycastle.org`.
20. Moritz Horsch. Mobile Authentisierung mit dem neuen Personalausweis (MONA). Master thesis, Technische Universität Darmstadt, July 2011.
21. J. Buchmann, A. Wiesmaier, D. Hühnlein, J. Braun, M. Horsch, F. Kiefer, and F. Strenzke. Towards a mobile eCard Client. In *Tagungsband zum 13. KryptoTag*, page 4, December 2010.
22. A. Wiesmaier, M. Horsch, J. Braun, F. Kiefer, D. Hühnlein, F. Strenzke, and J. Buchmann. An efficient mobile PACE implementation. In *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*, ASIACCS '11, pages 176–185, New York, NY, USA, March 2011. ACM.
23. J. Braun, M. Horsch, A. Wiesmaier, and D. Hühnlein. Mobile Authentisierung und Signatur. In Peter Schartner and Jürgen Taeger, editors, *D-A-CH Security 2011: Bestandsaufnahme, Konzepte, Anwendungen, Perspektiven*, pages 32–43. syssec Verlag, September 2011.
24. D. Hühnlein, D. Petrautzki, J. Schmölz, T. Wich, M. Horsch, T. Wieland, J. Eichholz, A. Wiesmaier, J. Braun, F. Feldmann, S. Potzernheim, J. Schwenk, C. Kahlo, A. Kühne, and H. Veit. On the design and implementation of the Open eCard App. In *GI SICHERHEIT 2012 Sicherheit - Schutz und Zuverlässigkeit*, 2012.