Password Authenticated Key Agreement for Contactless Smart Cards

Markus Ullmann[1], Dennis Kügler[1], Heike Neumann[2], Sebastian Stappert[2], and Matthias Vögeler[2]

[1] Bundesamt für Sicherheit in der Informationstechnik,
Markus.Ullmann@bsi.bund.de, Dennis.Kuegler@bsi.bund.de,
WWW home page: http://www.bsi.bund.de
[2] NXP Semiconductors,
Business Line Identification
Heike.Neumann@nxp.com, Sebastian.Stappert@nxp.com,
Matthias.Voegeler@nxp.com,
WWW home page: http://www.nxp.com

Abstract. This paper describes and compares the usage of password-based authenticated key agreement protocols to establish a secure communication channel between terminal and contactless card. In particular, protocols of this kind are discussed for use in contactless ID cards. The aim of this paper is to discuss, for the first time, two cryptographic password-based protocols with respect to security, implementation efforts and performance. Furthermore, a real life implementation on NXP's high security SmartMX chip is presented.

1 Introduction

1.1 Security Requirements

One significant security attack concerning contactless smart cards is the communication between an attacker's terminal and the card without the knowledge of the cardholder, while he carries the contactless card in his pocket. This attack is possible, even with passive contactless smart cards within the activation distance of the contactless card. In this context "passive" means that the smart card has no electrical power supply (e.g. battery). Regarding contactless smart cards (PICC) and terminals (PCD) according ISO/IEC 14443 [15–18], the real activation distance depends on technical quantities such as terminal power, terminal antenna coil and antenna diameters of terminal and card [12]. Measurement results are published in [11].

Besides that, an adversary might eavesdrop an existing radio frequency data transmission between terminal and contactless card. Again, in the case of contactless smart cards and terminals with an ISO/IEC 14443 interface the real range for eavesdropping a communication depends on technical quantities such as magnetic field strength, signal to noise ratio, noise class [13] etc. [12]. Considering noise level issues [13] the maximum range for eavesdropping the communication of a contactless smart card (PICC) is below 3 meters. This means, an adversary has to be relatively close with his antenna to successfully collect the communication data. To address the security risks mentioned, specific security mechanisms are needed. They have to fulfill following security requirements to resist these security threats:

– authentication of terminals
– strong session key agreement between authenticated terminal and contactless card (for the establishment of secure channels)
– forward secrecy of the session keys

The Basic Access Control Protocol (BAC) is the first approach which addresses the security requirements mentioned. This cryptographic protocol was developed (for first generation of e-passports, electronic travel documents containing facial images) to protect personal data from unauthorized access. The whole specifications of electronic travel documents are standardized by the International Civil Aviation Organization (ICAO), see [1–3]. On the one hand it describes how to implement data structures and commands. On the other hand it specifies the realization of authenticity, integrity and confidentiality of the electronic data stored on the radiofrequency chip embedded in the travel document. The access of a terminal to the contactless card itself and any data group on the card require at least the successful execution of the BAC protocol. A successful run of the BAC protocol itself requires knowledge of the Machine Readable Zone (MRZ), which is printed on the inner surface of the passport. The terminal needs this information to calculate the passport specific BAC authentication key to perform a successful BAC protocol run. Thus, without opening the passport, no data group can be read from a terminal.

Overall, the BAC protocol and its technical realization have some limitations:

– The entropy of the derived symmetric keys is in general less than 73 bits (in most cases even far below 56 bits) and thus does not prevent eavesdropping at all
– The BAC authentication key is static

Calculating the BAC authentication key from an eavedropped session is technically possible, but this still requires more effort than to obtain the personal data of the e-passport from other sources.

In order to overcome the first limitation, new password-based cryptographic protocols are discussed for an authenticated connection establishment between terminals and contactless cards. This idea goes back to advice of the BSI, first published as Password Authenticated Connection Establishment (PACE) in [9].

The second limitation is beyond the scope of this paper. This limitation and a technical solution are discussed in detail in [14].

1.2 Password-based Cryptographic Protocols

The basic idea of password-based cryptographic protocols is to combine a strong session key agreement with an implicit entity authentication based on a shared secret with limited entropy, called a password, in one cryptographic protocol. The initial idea goes back to Bellovin and Merret and their publication of the Encrypted Key Exchange protocol (EKE) [10]. Beyond the security requirements mentioned in subsection 1.1, password-based protocols themselves have to fulfill further security properties.

At first, if passwords with low entropy (e.g. passwords with 6 numeric characters) are considered, one boundary condition is the possibility that an adversary can search through all secret passwords in a reasonable time. Now, off-line and on-line dictionary attacks should not feasible.

1. Security against off-line dictionary attacks: A passive eavesdropper who records one or more protocol runs cannot get any information concerning password used for these protocol runs; in particular, he cannot calculate the password based on the protocol transcripts
2. Security against on-line dictionary attacks: There are two types of on-line dictionary attacks:
   - Type (1): An active adversary cannot abuse the protocol so as to eliminate a significant number of possible passwords (Partition Attacks).
   - Type (2): An active adversary can only test at most one password per protocol run by attempting to masquerade using this password (Brute-Force Attacks).

For example, EKE is vulnerable to on-line dictionary attack (1). Bellovin and Merrit introduced the notion of partition attacks against EKE. After enforcement of multiple decryption runs with distinct passwords, passwords can be separated into partitions of valid and invalid sets. Invalid passwords can be discarded. In order to make partition attacks harder, a lot of ideas were suggested. In succession, variants of password-based protocols were developed and discussed for client-server authentication since the first work of Bellovin and Merret. A good compendium on published password-based protocols is to be found on [19].

In this paper we present and discuss two different password-based protocols for the establishment of authenticated radio frequency connections between terminal and contactless card, the Password Authenticated Connection Establishment (PACE) [9] and TC-AMP. While PACE was designed for the establishment of authenticated channels between terminals and contactless cards, TC-AMP is a reduction of TP-AMP [4] intended for the mentioned purpose, and is first published and discussed within this paper.

Both fulfill the requirements mentioned in subsection 1.1, as well as being suitable for use with elliptic curve cryptography, which is the preferred approach for asymmetric cryptography on resource limited hardware such as contactless smart cards.

1.3 Structure of the Paper

The aim of section 2 is to describe the PACE protocol from different perspectives: protocol steps, implementation effort, performance and security. Here, only a security argumentation is given. A formal logic and cryptographic proof are beyond the scope of this paper. In section 3 the protocol TP-AMP is presented first. Moreover, this section presents a reduction of TP-AMP, called TC-AMP (TC is a shortcut for terminal and card), a brief security analysis and an implementation of TC-AMP. Finally, in section 5 a comparison of both protocols and hints concerning further studies are given.

1.4 Used Notation

Abbreviation	**Semantics**
$< G >$	cyclic group
G, G'	elliptic curve base point
$\Gamma, A, B, K, M, P, Q, T, X_1, X_2, Y_1, Y_2$	elliptic curve point
A_x, B_x, M_x, Q_x	x-coordinate of the curve point
π	shared short secret (password) with limited entropy
$\mathbb{Z}_n^*$	multiplicative group of n
x, x_1, x_2, y, y_1, y_2	random value $\in \mathbb{Z}_n^*$
s	random secret
k_i, sk_C, sk_S, μ	symmetric key
k_{Enc}	symmetric key for encryption
$h(), h_i()$	strong hash function
ENC()	symmetric encryption algorithm
DEC()	symmetric decryption algorithm
MAC()	Message Authentication Code calculation
k_{MAC}	symmetric key for MAC calculation
PCD	terminal
$PICC$	contactless smart card

Fig. 1. Abbreviations

The terms *terminal* and *reader* are used in this paper as synonyms.

2 Password Authenticated Connection Establishment (PACE)

2.1 Protocol Description

The PACE protocol, see [9], is adaptable for prime fields and elliptic curves. Here, in order to increase the performance we have chosen to use the elliptic curves variant.

The operations are then performed in the cyclic group $< G > := \{t * G | t \in \mathbb{N}\}$, $n := | < G > |$. In the following, $< G >^*$ denotes the cyclic group $< G >$ without the point at infinity. A practical method is the use of published secure domain parameter of a trusted authority, see [5].

The protocol consists of the following steps, the complete implementation is shown in Figure 2. Before the protocol starts the communication partners (terminal and smart card) of course have to agree on an elliptic curve E and a base point G.

1. The protocol starts with the selection of a random number s, with $0 \leq s < 2^m$, by the smart card in step (a). m is defined as the block size of the blockcipher used for the encryption of s. Both the smart card and the terminal derive a key μ using a key derivation function, here $h(\pi|1)$ is used. In the next step s is encrypted using a blockcipher with key μ, $z = \mathrm{ENC}(\mu, s)$, and z is then transmitted to the terminal, which decrypts z.
2. The terminal and the card map the nonce s to a new base point G'. Here the following randomized mapping is used:
   (a) An anonymous Diffie Hellman key agreement based on G is used to calculate a random point $P \in < G >^*$ (steps (e) - (i)).
   (b) Thereupon, P and s are exclusively used to calculate a new base point $G' = s*G+P$ in step (j) for the subsequent Diffie Hellman key agreement.
3. An anonymous Diffie-Hellman key agreement based on the new base point G' is performed to calculate a common secret curve point K (steps (k) - (o)).
4. Then, two different keys $k_{\mathrm{ENC}} = h(K_x|1)$ for encryption and $k_{\mathrm{MAC}} = h(K_x|2)$ for calculation of Message Authentication Codes (MAC) are derived from K. First, k_{MAC} is used for a MAC-calculation in step (p) and (q) performed as mutual authentication of terminal and card in steps ((r) - (u)).

After a successful PACE protocol run, Secure Messaging is started using the derived keys k_{ENC} and k_{MAC}.

2.2 Security of PACE

It is important to note that π is a static secret with low entropy. That means an adversary knows in principle the whole set of possible passwords π. The security of PACE strongly depends on the Computational Diffie-Hellman assumption and the secrecy of the password π. Assuming that K is calculated securely and erased after each PACE protocol run, PACE provides strong forward secrecy.

If the attacker were able to guess $\log_G G'$, PACE could be reduced to an anonymous Diffie-Hellman. As G' is calulated directly from s and P the choice of P has the following influence on the security:

- If a single, static point P is used, the discrete logarithm of G' can be easily guessed once $\log_G P$ is known. This allows an attacker to break a complete set of parameters used for PACE by calculating a single discrete logarithm. Furthermore, the distribution of G' in $< G >^*$ is very biased.

terminal (PCD)		**smart card (PICC)**	Step
		choose $0 \leq s < 2^m$ randomly	(a)
$\mu = h(\pi\|1) \bmod n$		$\mu = h(\pi\|1) \bmod n$	(b)
		$z = \mathrm{ENC}(\mu, s)$	(c)
	$\xleftarrow{z}$		
$s = \mathrm{DEC}(\mu, z)$			(d)
choose $x_1 \in \mathbb{Z}_n^*$ randomly			(e)
$X_1 = x_1 * G$			(f)
	$\xrightarrow{X_1}$		
		abort if $X_1 \notin < G >^*$	
		choose $y_1 \in \mathbb{Z}_n^*$ randomly	(g)
		$Y_1 = y_1 * G$	(h)
	$\xleftarrow{Y_1}$		
abort if $Y_1 \notin < G >^*$			
$P = x_1 * Y_1$		$P = y_1 * X_1$	(i)
$G' = s * G + P$		$G' = s * G + P$	(j)
choose $x_2 \in \mathbb{Z}_n^*$ randomly			(k)
$X_2 = x_2 * G'$			(l)
	$\xrightarrow{X_2}$		
		abort if $X_2 \notin < G >^*$	
		choose $y_2 \in \mathbb{Z}_n^*$ randomly	(m)
		$Y_2 = y_2 * G'$	(n)
	$\xleftarrow{Y_2}$		
abort if $Y_2 \notin < G >^*$			
$K = x_2 * Y_2$		$K = y_2 * X_2$	(o)
$k_{\mathrm{MAC}} = h(K_x\|2)$		$k_{\mathrm{MAC}} = h(K_x\|2)$	(p)
$t_{\mathrm{PCD}} = \mathrm{MAC}(k_{\mathrm{MAC}}, Y_{2,x})$			(q)
	$\xrightarrow{t_{PCD}}$		
		$t_{\mathrm{PICC}} = \mathrm{MAC}(k_{\mathrm{MAC}}, X_{2,x})$	(r)
	$\xleftarrow{t_{\mathrm{PICC}}}$		
		$t'_{\mathrm{PCD}} = \mathrm{MAC}(k_{\mathrm{MAC}}, Y_{2,x})$	(s)
		abort if $t'_{\mathrm{PCD}} \neq t_{\mathrm{PCD}}$	(t)
$t'_{\mathrm{PICC}} = \mathrm{MAC}(k_{\mathrm{MAC}}, X_{2,x})$			(u)
abort if $t'_{\mathrm{PICC}} \neq t_{\mathrm{PICC}}$			(v)

Fig. 2. PACE

– If a fresh point P is calculated in every protocol run, it is guaranteed that the discrete logarithm of G' is unknown to both parties. If at least one party chooses its secret value (i.e. x_1 or x_2, respectively) randomly and uniformly, P and thus G' are distributed randomly and uniformly in $< G >^*$.

To summarize, while the randomized mapping is more complex, it has some advantages in terms of security. Finally, it is sketched how PACE prevents the attacks introduced in Section 1.2:

Off-line Dictionary Attacks: If the nonce s is chosen s.th. it is randomly and uniformly distributed in $\{0, \ldots, 2^m - 1\}$, z is also randomly an uniformly distributed in $\{0, \ldots, 2^m - 1\}$ as the password-based encryption is a pseudo-random permutation. Thus, the attacker cannot extract any information on the password from eavesdropping.

On-line Dictionary Attacks: The attacks are prevented as follows:

- **Type (1):** Due to the distribution of s and z the attacker cannot exclude passwords, as every password maps s to a value z with equal probability. Thus, Partition Attacks are impossible.
- **Type (2):** If the blockcipher is a pseudo-random permutation, the attacker is restricted to test exactly one password per interaction. Thus, Brute-Force Attacks are restricted to test one password per protocol run.

2.3 Implementation of PACE

This section describes details of the implementation of the PACE protocol.

For the performance analysis described in subsection 2.4 a PACE implementation on NXP's smart card controller SmartMX was developed. This implementation is based on NXP's certified crypto library on the SmartMX. This section contains, beside the description of the basic crypto functionality needed, the ISO/IEC 7816 command mapping.

For the implementation of PACE the elliptic curve domain parameters brainpoolP224r1 from [5] have been chosen. For the encryption of the random value s using key μ a 3-DES encryption in ECB-Mode was used (cf. [8]).

The implementation of PACE uses the following basic cryptographic functionality:

1. EC key generation in step (j) of figure 5 for $s * G$,
2. EC Diffie-Hellman key exchange (also used for elliptic curve scalar point multiplication) in steps (h), (i), (n) and (o),
3. EC point addition in step (j),
4. 3-DES encryption, ECB/CBC mode in steps (c), (r) and (s).

The cryptographic functions for the native implementation have been written in the programming languages C and Assembler, the implementation is analyzed in the next section.

ISO 7816 Commands The following sequence of ISO 7816 commands is used to implement PACE:

1. MSE:Set AT
2. General Authenticate
   (a) Encrypted Nonce
   (b) Map Nonce
   (c) Perform Mutual Key Agreement
   (d) Mutual Authenticate

The command General Authenticate is divided into the above four sub-commands. Table 1 provides the ISO/IEC 7816 command mapping that has been used, for more details see also [9].

Table 1. ISO/IEC 7816 Command Mapping

MSE: Set AT			
	CLA/INS/P1/P2	Data	LE
C-APDU	00 22 C1 A4	80 01 11 83 01 02	—
	SW	Data	
R-APDU	90 00	—	
General Authenticate: Encrypted Nonce			
	CLA/INS/P1/P2	Data	LE
C-APDU	10 86 00 00		22
	SW	Data	
R-APDU	90 00	80 20 Encrypted Nonce	
General Authenticate: Map Nonce			
	CLA/INS/P1/P2	Data	LE
C-APDU	10 86 00 00	81 39 04 Ephemeral Public Key	3B
	SW	Data	
R-APDU	90 00	82 39 04 Ephemeral Public Key	
General Authenticate: Perform Mutual Key Agreement			
	CLA/INS/P1/P2	Data	LE
C-APDU	10 86 00 00	83 39 04 Ephemeral Public Key	3B
	SW	Data	
R-APDU	90 00	84 39 04 Ephemeral Public Key	
General Authenticate: Mutual Authenticate			
	CLA/INS/P1/P2	Data	LE
C-APDU	00 86 00 00	85 08 Authentication Token	0A
	SW	Data	
R-APDU	90 00	86 08 Authentication Token	

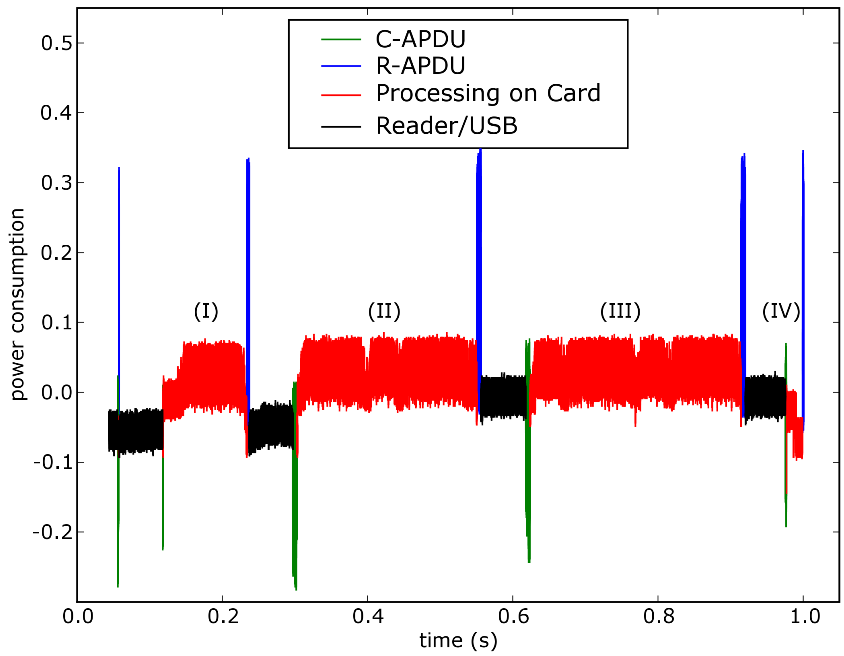

Fig. 3. Processing of the PACE protocol

2.4 Performance Analysis of PACE

For the results in this section a measurement setup as described in appendix A has been used.

Figure 3 shows the execution of the PACE protocol as it has been described in section 2.1 on NXP's SmartMX. As a whole the protocol consists of 5 Command- and Response-APDUs.

The green peaks pointing down indicate communication from PCD to PICC, the blue peaks pointing up indicate communication from PICC to PCD. So between a green and a blue peak the smart card is processing a Command-APDU.

In figure 3 four phases are marked with Roman numerals (I) to (IV). The next description gives the relation between the outline of PACE from figure 2 and the real implementation shown visually in figure 3.

Phase (I) In this phase the smart card performs steps (a) to (c) and the first part of step (j). Step (a) and the first part of (j) is just realized by performing an elliptic curve key generation. Note, that step (b) can be omitted if a static password π is used.

Phase (II) In the second phase the smart card performs steps (g) and (h) by doing an elliptic curve key generation, and step (i) by performing an elliptic curve Diffie-Hellman key exchange, and finally the second part of step (j) is computed the elliptic curve point addition.

Phase (III) In this phase the smart card computes steps (m) to (o) by doing two elliptic curve Diffie-Hellman key exchanges. Also, step (p), the key derivation, is performed.

Phase (IV) Finally only steps (r) to (t) are performed, i.e. computing and checking the MACs.

The total execution time of the PACE protocol is about 945 ms, where the time consumption for the contactless communication is only about 32 ms (3.3%). Most of the time, about 677 ms (71.6%), is used for the operations on the smart card.

The power consumption increases significantly when the FameXE, the co-processor for asymmetric cryptography on the SmartMX, is used. It is possible to detect the two elliptic curve key generations and three elliptic curve scalar point multiplications, which are performed by PACE, in figure 3.

3 TP-AMP

In [4] T. Kwon introduced a new efficient three-pass password authenticated key agreement protocol which is proven to be secure under the Diffie-Hellman intractability assumption in the random-oracle model. This protocol is discussed for the use in general server/client environments and the author also pointed out that this new protocol was contrived to resist server compromise.

For the protocol, the client (C) and Server (S) have to agree on Diffie-Hellman key agreement parameters, i.e. on primes p, q, where q divides $p-1$ and on an element g with order q in $\mathbb{Z}^*$. Let $h_i(.) = h(i|.|i)$, where h is a strong one-way hash function, i is an integer. Furthermore, let $H_i(.) = \zeta^{h_i(.)} \bmod p$ with another generator ζ.

In the registration phase a user chooses a name C and a password π which are stored in the server as well. Then for the protocol the server and the client derive some values from these parameters: $\gamma = H_0(C|\pi)$, $\gamma' = \gamma^{-1} \bmod p$, $u = h_1(C|\pi)$ and $\nu = g^u \bmod p$. If the protocol as described in figure 4 has been successfully executed, both parties derive session keys, the server computes sk_S and the client sk_C.

Figure 4 shows the execution of the protocol.

It is obvious that α and β correspond to the shared secret that is computed on both sides of the protocol. For cryptographic strong one-way hash functions h it is very likely that the protocol succeeds if and only if $\alpha = \beta$, otherwise with a very high probability the exchanged hash values will be different.

The author of [4] also addresses that for arbitrary client/server environments, where several clients are able to connect in parallel to a server, guessing attacks

client gets (C, π)		**server on** S
$\gamma = H_0(C\|\pi)$, $\gamma' = \gamma^{-1} \bmod p$		$\gamma = H_0(C\|\pi)$, $\gamma' = \gamma^{-1} \bmod p$ $\nu = g^u \bmod p$ with $u = h_1(C\|\pi)$
choose x in $\mathbb{Z}_q^*$ randomly $m = g^x \gamma \bmod p$		
	$\xrightarrow{C, m}$	
		abort if there is no account for C choose y in $\mathbb{Z}_q^*$ randomly $\mu = \nu^y \bmod p$ $\beta = (m\gamma' g^m)^y \bmod p$ $k_1 = h_2(C\|S\|m\|\mu\|\beta\|\gamma')$
	$\xleftarrow{\mu, k_1}$	
$w = u^{-1}(x + m) \bmod q$ $\alpha = \mu^w \bmod p$ $k_1' = h_2(C\|S\|m\|\mu\|\alpha\|\gamma')$ abort if $k_1 \neq k_1'$ $k_2 = h_3(C\|S\|m\|\mu\|\alpha\|\gamma')$		
	$\xrightarrow{k_2}$	
		$k_2' = h_3(C\|S\|m\|\mu\|\beta\|\gamma')$ abort if $k_2 \neq k_2'$
$sk_C = h_4(C\|S\|m\|\mu\|\alpha\|\gamma')$		$sk_S = h_4(C\|S\|m\|\mu\|\beta\|\gamma')$

Fig. 4. TP-AMP

might be a problem. In the next section we point out that we need not take care of this situation for our communication setup.

3.1 Adaption of TP-AMP

In this paper we want to simplify the protocol TP-AMP for the use as an authentication protocol on contactless operating smart cards. In this scenario the smart card acts as the server and a reader operates as the client. The ISO/IEC 14443 standard ensures that a card, which acts here as the server, communicates at most with one reader.

Furthermore, in order to increase the performance we want to adapt the protocol such that it uses elliptic curves instead of prime fields, we call this protocol then TC-AMP.

The communication partners of course have to agree on an elliptic curve E and base point G. The operations are then performed in the cyclic group $< G >:= \{t * G | t \in \mathbb{N}\}$, $n := | < G > |$. In the following, $< G >^*$ denotes the cyclic group $< G >$ without the point at infinity and for every point M of the elliptic curve M_x denotes the x-coordinate of the point M. In addition to that we need a second base point $G' \in < G >^*$ which is chosen so that no $l \in \mathbb{Z}_n^*$ is known with $G' = l * G$.

Note: For the implementation of PACE a randomized mapping was used where the second generator G' was calculated by an anonymous Diffie-Hellmann. For reasons of efficiency, TC-AMP is implemented with a static mapping and therefore, a fixed second static base point must be made available in the domain parameter of the elliptic curve. In this case it is required that $\log_G G'$ is unknown.

Figure 5 gives the procedure of TC-AMP. It consists of the following steps:

1. From the password π the values $u = h_0(\pi) \bmod n$, $\Gamma_0 = u*G$, and $\Gamma_1 = u*G'$, $\Gamma_1' = -\Gamma_1$ are derived.[3]
2. The terminal computes a random point M on the elliptic curve and sends it to the chip, which then computes a random point Q on the elliptic curve and sends it to the terminal (step (c)-(f)). While M depends on the secret Γ_1, Q depends on the secret Γ_0.
3. Furthermore, the chip computes the common secret B in step (e) and derives in step (f) the hash value k_1 of the x-coordinates of M, Q and B. The hash value k_1 is sent to the terminal, which uses Q to compute the common secret A. If both parties know the correct value of Γ_0 and Γ_1 it follows that $A = B$ so that the terminal is able to verify k_1 in steps (j) and (k). The terminal then computes a different hash value k_2 of the x-coordinates of M, Q and A and sends it to the PICC, which is then able to verify this hash value if $A = B$ in steps (m) and (n). Finally both parties derive session keys for secure messaging in step (o).

terminal (PCD) gets π		**smart card (PICC)**	Step
$u = h_0(\pi) \bmod n$, $\Gamma_0 = u * G$, $\Gamma_1 = u * G'$, $\Gamma_1' = -\Gamma_1$		$u = h_0(\pi) \bmod n$, $\Gamma_0 = u * G$, $\Gamma_1 = u * G'$,	(a)
choose $x \in \mathbb{Z}_n^*$ randomly such that $M = (x * G) + \Gamma_1' \in < G >^*$ and $m = M_x \bmod n$			(b)
	$\xrightarrow{M}$		
		abort if $M \notin < G >^*$ $m = M.x \bmod n$	(c)
		choose $y \in \mathbb{Z}_n^*$ randomly such that $Q = y * \Gamma_0$	(d)
		$B = y * (M + \Gamma_1 + (m * G))$	(e)
		$k_1 = h_2(M_x\|Q_x\|B_x)$	(f)
	$\xleftarrow{Q, k_1}$		
abort if $Q \notin < G >^*$			(g)
$w = u^{-1}(x + m) \bmod n$			(h)
$A = w * Q$			(i)
$k_1' = h_2(M_x\|Q_x\|A_x)$			(j)
abort if $k_1 \neq k_1'$			(k)
$k_2 = h_3(M_x\|Q_x\|A_x)$			(l)
	$\xrightarrow{k_2}$		
		$k_2' = h_3(M_x\|Q_x\|B_x)$	(m)
		abort if $k_2 \neq k_2'$	(n)
$k_{\text{ENC}} = h(A_x\|1)$, $k_{\text{MAC}} = h(A_x\|2)$		$k_{\text{ENC}} = h(B_x\|1)$, $k_{\text{MAC}} = h(B_x\|2)$	(o)

Fig. 5. TC-AMP

In order to adapt TP-AMP to our needs we carry out the following simplifications.

A smart card operating according to the ISO/IEC 14443 standard does not need the identifiers S and C in the protocol as for TP-AMP, since a card, which acts here as a server, can only communicate at most with one reader.

Furthermore we omit values Γ_0 and Γ_1 from hashing since the values M and Q directly depend on Γ_0 or Γ_1. Please note, that it is not advisable to simplify the protocol further so that M is computed as $x * G$ and sent, such that B does not depend on Γ_1. If the protocol is simplified in this way, it is highly vulnerable to an off-line dictionary attack, where the attacker only needs one response (Q, k_1) in order to mount the attack.

3.2 Security of TC-AMP

The security of the TC-AMP protocol relies on the intractability assumption of the discrete logarithmic problem, the cryptographic strength of the hash function and the secrecy of the password π.

A Man-in-the-Middle attack is only possible if the adversary knows the password π since all hash values k_i indirectly depend on the knowledge of π.

If the adversary is the reader, he can freely choose the value M in step (b) in figure 5 and collect responses (Q, k_1). First of all the problem to find an M such that $m * G = -M$ is intractable or even unsolvable. Next, in order to prevent small subgroups attacks, the smart card has to check that $M \in < G >^*$. Since y is a random value it is not disclosed by Q. On the other hand Q gives a hint for $(y * G)$ because of the low entropy of π there are only a few u, i.e. if u' is a guess for u than $u'^{-1} * Q$ is a guess for $(y * G)$. It is important that B does not linearly depend on G if the adversary does not know Γ_1, that is why no linear relationship between G' and G shall be known. The adversary can use the value k_1 for an exhaustive search for B. Although in case he reveals B he can successfully proceed the protocol, π is still not disclosed for upcoming sessions.

If the adversary is a smart card, he is able to collect M. Since x is chosen randomly, no information about π is disclosed. The first response (Q, k_1) of the adversary will reveal that he does not know the password π.

Note, in step (e) of the protocol in figure 5 the elliptic curve point B can be equal to the point at infinity. Though the probability of this case is very low we do not recommend to abort the protocol since an adversary observing the execution of the protocol would be able to use the relation $\Gamma_1 = -M - m * G = h_0(\pi) * G'$ for an exhaustive search for π. Instead of aborting the protocol one just sets B_x equal to a predefined constant.

As a matter of principle elliptic curves with co-factor equal to 1 should be used in order to exclude small-subgroup attacks. The security environment should only use cryptographically secure elliptic curves where the parameter generation can be reproduced. For TC-AMP, one particularly needs to understand

[3] For a real-life security environment one has to ensure that $h_0(\pi) \not\equiv 0 \pmod{n}$ for all π.

that no linear relationship between the base points G and G' is known. As an example the derivation of the Brainpool elliptic curves [5] is most transparent.

We pointed out that we do not directly use the secrets Γ_0 or Γ_1 for hashing as it was considered for TP-AMP. For a static π, the elliptic points Γ_0 and Γ_1 are also static and might be subject to a side channel attack. Since it is hard to secure standard hash algorithms against side channel attacks it is then even an advantage to omit Γ_0 or Γ_1 from hashing.

Referring to the attacks listed on page 3, firstly, an adversary is not able to perform an off-line dictionary attack since M and Q are random points on the elliptic curve and even if he succeeds to derive A or B from the hash values k_1 or k_2 he is not able to derive π. Secondly, an on-line dictionary attack (1), where an active adversary is able to exclude several passwords from one protocol run, is not possible provided that the mapping $\pi \mapsto \Gamma_1$ is injective, since there exists only one Γ_1 such that the protocol aborts in step (f) and the two exchanged hash values do not reveal any practical information about π. Of course an adversary is able to perform an on-line attack (2).

3.3 Implementation of TC-AMP

This section describes details of the implementation of the TC-AMP protocol.

For the performance analysis within the next section a TC-AMP implementation on NXP's smart card controller SmartMX was developed. This implementation is based on NXP's certified crypto library on the SmartMX. This section contains beside the description of the basic crypto functionality needed the ISO/IEC 7816 command mapping.

For the implementation of TC-AMP the elliptic curve domain parameters brainpoolP224r1 from [5] have been chosen. For the hash function h_i the hash algorithm SHA-1 was used (see [6]).

The implementation of TC-AMP uses the following basic cryptographic functionality:

1. EC Diffie-Hellman key exchange used as elliptic curve scalar point multiplication in step (a), (d) and twice in step (e) of figure 5,
2. EC point addition twice in step (e),
3. SHA-1 hash function in steps (a), (g), (m) and (o),
4. Modular reduction in step (a) and for m in step (e).

The cryptographic functions for the native implementation are written in the programming-languages C and Assembler and the implementation is analyzed in the next section 3.4.

ISO/IEC 7816 Commands The following sequence of commands is used to implement TC-AMP:

1. MSE:Set AT
2. General Authenticate

(a) Key Agreement
(b) Mutual Authenticate

The command General Authenticate is divided into the above two sub-commands by using the ISO/IEC 7816 command chaining.

Table 2 provides the ISO/IEC 7816 command mapping that has been used.

Table 2. ISO/IEC 7816 Command Mapping

MSE: Set AT			
	CLA/INS/P1/P2	Data	LE
C-APDU	00 22 C1 A4	80 01 12 83 01 02	—
	SW	Data	
R-APDU	90 00	—	
General Authenticate: Key Agreement			
	CLA/INS/P1/P2	Data	LE
C-APDU	10 86 00 00	80 39 04 M_xM_y	4F
	SW	Data	
R-APDU	90 00	81 4D 04 $Q_xQ_yk_1$	
General Authenticate: Mutual Authenticate			
	CLA/INS/P1/P2	Data	LE
C-APDU	00 86 00 00	82 14 k_3	—
	SW	Data	
R-APDU	90 00	—	

3.4 Performance Analysis of TC-AMP

For the results in this section a measurement setup as described in appendix A have been used.

Figure 6 shows the execution of the TC-AMP protocol on the NXP's SmartMX as it has been described in figure 5. As a whole the protocol consists of 3 Command- and Response-APDUs.

The green peaks pointing down indicate communication from PCD to PICC, the blue peaks pointing up indicate communication from PICC to PCD. So between a green and a blue peak the smart card is processing a Command-APDU.

In figure 6 we marked six phases with Roman numerals (I) to (VI). The next description gives the relation between the outline of TC-AMP from figure 5 and the real implementation shown visually in figure 6.

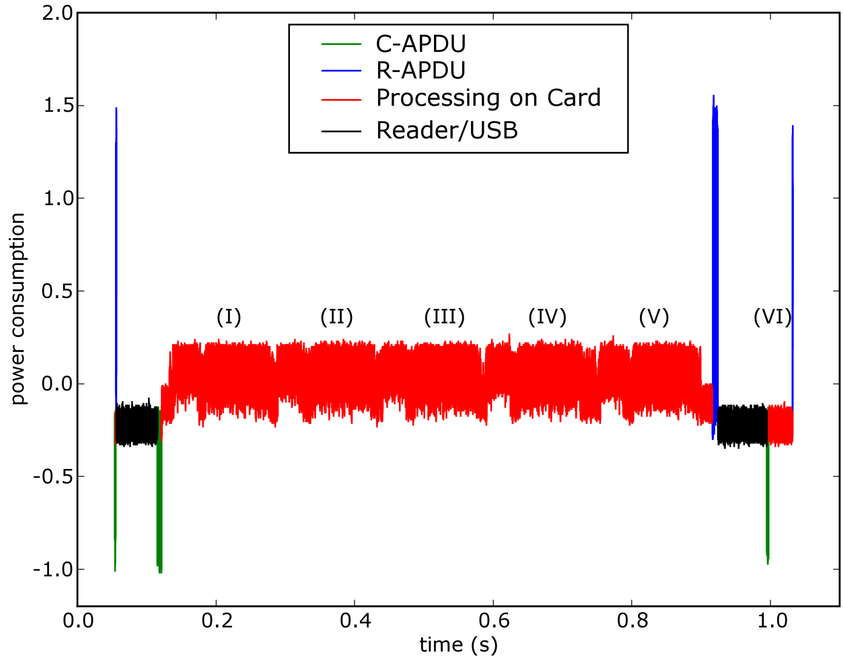

Fig. 6. Processing of the TC-AMP protocol

Phase (I) In this phase the smart card computes Γ_0 from step (a) by performing an elliptic curve scalar point multiplication. Note, that this phase can be omitted if a static password π is used.
Phase (II) In the second phase the smart card performs step (d) by doing an elliptic curve key generation and step (i) by performing an elliptic curve scalar point multiplication.
Phase (III) In this phase the smart card computes Γ_1 from step (a) by doing an elliptic curve scalar point multiplication. Note, that this phase can be omitted if a static password π is used.
Phase (IV) In this phase the smart card performs steps (c) and $m*G$ by doing an elliptic curve scalar point multiplication.
Phase (V) Finally in the second APDU the remaining operations of step (e) and steps (f), (g) are performed.
Phase (VI) In the last phase the value k_2' from step (m) is computed.

The total execution time of the TC-AMP protocol is about 978 ms, where the time consumption for the contactless communication is only about 18 ms (1.9%). Most of the time, about 830 ms (84.8%), is used for the operations on the smart card.

The power consumption increases significantly if the FameXE, the co-processor for asymmetric cryptography on the SmartMX, is used. One elliptic curve key generation and three elliptic curve scalar point multiplications, which are needed by TC-AMP, are visible in figure 6. The time needed by the four SHA-1 operations is negligible compared to the elliptic curve operations.

4 Dictionary Attacks on Passwords

Due to the low entropy of the used passwords, brute-force attacks cannot be ignored, especially in the contactless setting. In order to prevent such attacks a smart card operating system has the following options:

1. It can implement an error counter. If the error counter then reaches a certain value the smart card operating system permanently disables the smart card. For an contactless operating smart card the protocol might be executed by an attacker if he is able to place at close range a smart card reader. This attacker might just have the goal to disable the smart card. Thus simply increasing an error counter could be the wrong countermeasure against denial of service (DoS) attacks.
2. It can implement a time delay between a session where the protocol failed and a new session so that a guessing attack would consume too much time. If the time delay on the other hand is too big a DoS attack would be possible again. A standard technique for realizing a time delay is to use an EEPROM memory cell to realize a delay counter. Of course on the one hand that would stress the lifetime of a dedicated EEPROM cell a lot and on the other hand even more important, an adversary might be able to detect the EEPROM-writing routine and disable the power supply for the smart card before the

EEPROM-writing has been finished. The SmartMX of NXP offers a clever hardware solution that does not involve the usage of EEPROM, here, the smart card operating system is able to set a dedicated bit, called Delay Latch, which will be automatically erased again after a couple of minutes independently of the power supply of the smart card.

5 Conclusion

In this paper, we describe specific attacks concerning contactless smart cards. Furthermore, we introduce password-based authenticated key agreement protocols as countermeasures against unauthorized communication with the card and against eavesdropping of the data transmission between terminal and card. In particular, two different password-based protocols are presented and analysed, PACE and TC-AMP. PACE was designed for the establishment of authenticated channels between terminal and card. However, TC-AMP is a reduced variant of TP-AMP [4] and is first published in this paper. Both are suitable for elliptic curves. Here, both protocols are adapted to use elliptic curves. Therefore, a mapping function is necessary that maps a nonce (PACE) or the password π to a point on the elliptic curve. We have chosen to use a different mapping for PACE and TC-AMP (although the same could be used). PACE uses a more complex randomized mapping that has some security advantages and TC-AMP uses a simpler static mapping.

As a consequence, only three ISO 7816 commands are necessary to implement TC-AMP but five ISO 7816 commands are required to implement this variant of PACE. Although there is a large difference concerning the required APDUs, the performance is very similar. The reason is the extensive use of time-consuming EC scalar point multiplications in TC-AMP.

Implementations of PACE and TC-AMP on a native smart card operating system are the basis for real performance measurements. Within this paper, hints to Javacard implementations of PACE and TC-AMP are given although, until now, no such JCOP implementations are available; this will be addressed within future work.

Furthermore, formal cryptographic proofs of security for PACE and TC-AMP as yet do not exist; these are the subject of current studies.

A Measurement Setup

The measurement setup for the performance analysis is sketched in figure 7. A host computer controls a contactless card reader, in our case the NXP Pegoda Reader, to perform the cryptographic algorithm on the smart card. The contactless card reader generates a magnetic field that is used as a physical carrier for the power transfer and the data transmission. The smart card is powered by the magnetic field only. The smart card coil converts the magnetic field into a voltage that supplies the integrated circuit.

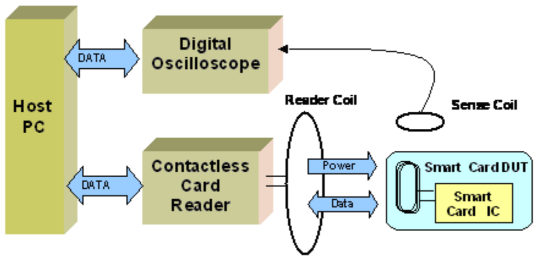

Fig. 7. Contactless Measurement Setup

A sense coil is located close to the smart card coil. It measures the magnetic field generated by the reader device and the superposed magnetic field caused by the smart card current. Assuming that the reader field has constant amplitude during the execution of the cryptographic functions, the AC amplitude of sense coil voltage presents the variations of the smart card current. An AM demodulator (peak detector) converts RF voltage to an amplitude signal. Because the usual attacks are independent of the DC component, the AC component of the amplitude voltage is perfect as input signal for a software analysis. A digital scope converts the analog measured coil voltage to digital data. The data are transferred to a host computer and stored on a disk drive.

The communication speed for the NXP Pegoda Reader and the smart card has been set to 106 kBaud/s.

B Notes for a Javacard Implementation

This section discusses an implementation of the PACE - and TC-AMP protocol on a Javacard operating system.

B.1 Remarks for PACE

For an implementation of the PACE protocol on a Javacard Operating System the following standard APIs (Javacard Version 2.2.2, cf. [7]) can be used:

– EC key generation (Class: KeyBuilder, Objects: TYPE_EC_FP_PRIVATE, TYPE_EC_FP_PUBLIC),
– 3-DES encryption, ECB mode (Class: Cipher, Object: ALG_DES_ECB_NOPAD).

In addition the following non-standard APIs which are not covered by Javacard API Version 2.2.2, cf. [7], are needed to implement PACE:

– EC point addition,
– EC Diffie-Hellman key exchange (without key derivation).

The standard Diffie-Hellman key exchange of the Javacard 2.2.2 API (Class: KeyAgreement) cannot be used because it includes the key derivation function (i.e. hashing of the result).

B.2 Remarks for TC-AMP

For an implementation of the TC-AMP protocol on a Javacard Operating System the following standard APIs (Javacard Version 2.2.2, cf. [7]) can be used:

- EC key generation (Class: KeyBuilder, Objects: TYPE_EC_FP_PRIVATE, TYPE_EC_FP_PUBLIC)
- SHA-1 hash function (Class: MessageDigest, Object: ALG_SHA)

Additionally the following non-standard APIs, which are not covered by Javacard API Version 2.2.2, cf. [7], are needed to implement TC-AMP:

- EC point addition,
- EC Diffie-Hellman key exchange,
- Modular reduction.

The standard Diffie-Hellman key exchange of the Javacard 2.2.2 API (Class: KeyAgreement) cannot be used because it includes the key derivation function (i.e. hashing of the result).

B.3 JCOP

NXP offers the Javacard Operating System JCOP on the SmartMX, which will have all pre-requisites for the implementation of PACE and TC-AMP.

References

1. ICAO Doc 9303. Part 1 - Machine Readable Passport. ICAO, 6th edition, 2006.
2. ICAO Doc 9303. Part 2 - Machine Readable Visas. ICAO, 3rd edition, 2005.
3. ICAO Doc 9303. Part 3 - Size 1 and Size 2 Machine Readable Official Travel Documents. ICAO, 2nd edition, 2002.
4. Kwon, T.: Practical Authenticated Key Agreement using Passwords. Springer Berlin / Heidelberg, LNCS **3225** (2004) 1–12
5. ECC Brainpool Standard Curves and Curve Generation, Version 1.0 (available online at http://www.ecc-brainpool.org/ecc-standard.htm), 2005
6. FIPS PUB 180-2, Secure Hash Standard, Federal Information Processing Standards Publication, August 1st, 2002, (plus Change Notice 1 to include SHA-224, February 25th, 2004), US Department of Commerce/National Institute of Standards and Technology
7. Java Card Platform: Application Programming Interface, Version 2.2.2 (available online at http://java.sun.com/products/javacard/specs.html), 2006
8. FIPS 46-3, Data Encryption Standard (DES), October 1999
9. BSI, Technical Guideline TR-03110: Advanced Security Mechanisms for Machine Readable Travel Documents Extended Access Control (EAC) and Password Authenticated Connection Establishment (PACE), Version 2.0, February 2008
10. Steven M. Bellovin and Michael Merritt, Augmented encrypted key exchange: Password-based protocol secure against dictionary attacks, Symposium on Research in Security and Privacy, IEEE Computer Society Press, 1992

11. Ziv Kfir and Avishai Wool, Picking Virtual Pockets using Relay Attacks on Contact-less Smartcard Systems, Proceedings of the First International Conference on Security and Privacy for Emerging Areas in Communication Networks, IEEE Computer Society Press, 2005
12. NXP, ISO/IEC 14443 Eavesdropping and Activation Distance, 13,56 MHz proximity smart cards, Application note, Rev. 01.01, Januar 2008
13. European Radiocommunication Committee (ERC) within the European Conference of Postal and Telecommunications Administrations, Propagation Model and Interference Range Calculation for Inductive Systems 10 kHz - 30 MHz, Marbella, February 1999
14. Markus Ullmann, Flexible Visual Display Units as Security Enforcing Component for Contactless Smart Card Systems, Proceedings of the First International EURASIP Workshop on RFID, Vienna 2007
15. ISO/IEC 14443, Contactless Integrated circuit(s) cards, Part 1: Physical Characteristics, April 2000
16. ISO/IEC 14443, Contactless Integrated circuit(s) cards, Part 2: Radios Frequency Power and Signal Interface, July 2001
17. ISO/IEC 14443, Contactless Integrated circuit(s) cards, Part 3: Initialization and Anticollision, February 2001
18. ISO/IEC 14443, Contactless Integrated circuit(s) cards, Part 4: Transmission Protocol, February 2001
19. David Jablon, List of Research Paper on Password-based Cryptography, (available online at www.jablon.org/passwordlinks.html), 2008