Supplemental Access Control (PACE v2): Security Analysis of PACE Integrated Mapping

Jean-Sébastien Coron[1], Aline Gouget[2], Thomas Icart[1], and Pascal Paillier[2]

[1] Université du Luxembourg
6, rue Richard Coudenhove-Kalergi, L-1359 Luxembourg, Luxembourg
jean-sebastien.coron@uni.lu, thomas.icart@m4x.org
[2] Gemalto
6, rue de la Verrerie, 92190 Meudon, France
aline.gouget@gemalto.com, pascal.paillier@gemalto.com

Abstract. We describe and analyze the password-based key establishment protocol PACE v2 Integrated Mapping (IM), an evolution of PACE v1 jointly proposed by Gemalto and Sagem Sécurité. PACE v2 IM enjoys the following properties:
- patent-freeness[3] (to the best of current knowledge in the field);
- full resistance to dictionary attacks, secrecy and forward secrecy in the security model agreed upon by the CEN TC224 WG16 group;
- optimal performances.

The PACE v2 IM protocol is intended to provide an alternative to the German PACE v1 protocol, which is also the German PACE v2 Generic Mapping (GM) protocol, proposed by the German Federal Office for Information Security (BSI). In this document, we provide
- a description of PACE v2 IM,
- a description of the security requirements one expects from a password-based key establishment protocol in order to support secure applications,
- a security proof of PACE v2 IM in the so-called Bellare-Pointcheval-Rogaway (BPR) security model.

1 Introduction

Password-Based Key Exchange. A password-based key exchange protocol enables two parties who share a low entropy password π to communicate securely over an insecure channel. Numerous such protocols have been published in the literature; see for example [3,4,14]. In this document we consider a scenario in which two entities - a chip A and a terminal B - wish to communicate securely. Initially the chip A holds a password, which is transmitted to terminal B by physical means; a typical scenario is to have chip A embedded in a passport, and the password written in the passport's Machine Readable Zone (MRZ) that is read by the terminal; alternatively, the password π may be a PIN code in contactless applications. The chip and the terminal would like to engage in an exchange of messages at the end of which each holds a session key sk, which is only known by the two of them. This session key can in turn be used in a secure messaging scheme in order to transmit some confidential information σ from the chip to the terminal.

Since the password can be of limited size, it may be possible for the adversary to enumerate offline all possible passwords from a dictionary $\mathcal{D}$; this is called a *dictionary attack*. We would like that the adversary's chance in learning the session key sk will depend only on her online interaction with the chip, and that it doesn't significantly depend on her off-line computing time. In other words, the goal of a password-based key exchange protocol is to make sure that the adversary cannot be more successful than by simply randomly guessing the password during the online phase. The system can then limit the number of interactions so that a password can be invalidated after a certain number of failed attempts.

A general security model for password-based key exchange has been defined by Bellare, Pointcheval and Rogaway in [4]. In this model, one considers a set of clients $C \in \mathcal{C}$ and a set of servers $S \in \mathcal{S}$; a client C

[3] PACE v2 IM relies on a cryptographic method (Icart's point encoding) owned by Sagem Sécurité. However Sagem Sécurité has agreed to grant free-of-charge exploitation rights under restrictive conditions [1].

and a server S communicate via a client instance C^i and a server instance S^j. In the concurrent model, the adversary can create several instances of a participant that can be simultaneously active, whereas in the non-concurrent model only one active user instance U^i is allowed between user U and server S. The adversary is granted access to plenty of honest executions; namely only passive eavesdropping is required to access them. Moreover the adversary can request session keys between some user instance U^i and server instance S^j; the loss of a session key should not compromise the security of other sessions. Eventually the adversary tries to tell apart a real session key used by U^{i^*} and S^{j^*} (not revealed) from a random one. In this model the adversary can also corrupt users to obtain their passwords; then previously established session-keys should remain secure; this is the forward secrecy property.

PACE v2 Integrated Mappping (IM). The Supplemental Acces Control for Machine Readable Travel Documents PACE v2 is defined in [11]. In this paper we describe and analyze the password-based key establishment protocol PACE v2 IM, an evolution of PACE v1 jointly proposed by Gemalto and Sagem Sécurité. The PACE v2 IM protocol is intended to provide an alternative to the German PACE v1 protocol, which is also called PACE v2 Generic Mapping (GM) protocol, proposed by the German Federal Office for Information Security (BSI). PACE v2 IM enjoys optimal performances and a maximal security level.

The goal of the PACE v2 IM protocol is that a chip and terminal establish a secure channel from a common secret password π with low and given entropy. More precisely, the chip and the terminal should generate a high-entropy ephemeral secret key material (one-time session keys and initialization vectors for secure messaging). PACE v2 IM relies on a list of cryptographic primitives and is generic towards the choice of these primitives. However, for PACE v2 IM to be securely implemented, it is important that the primitives be carefully chosen. At least one cryptographic suite shall be provided for secure implementations of PACE v2 IM. PACE v2 IM is based on elliptic curves and is generic towards the choice of the elliptic curve; the only restriction is that we must have $p = 2 \bmod 3$ where p is the characteristic of the prime field; this is because PACE v2 IM uses Icart's encoding function into elliptic-curve [12]. Alternatively PACE v2 IM can use the simplified SWU encoding function into elliptic-curve [9]. We provide a detailed description of PACE v2 IM in Section 2.

Security in a restricted Model. For simplicity we first analyze the security of PACE v2 IM in a restricted model of security, which can be seen as a simplification of the general model of [4]:

1. The chip cannot interact with more than one terminal at the same time, and the terminal cannot interact with more than one chip at the same time (non-concurrent model).
2. The terminal does not hold any secret, except the chip's password.
3. The adversary cannot tamper with the messages sent by the chip and the terminal during their interaction; it can only eavesdrop their communications. In particular, the adversary cannot interact with a terminal holding a password; however, the adversary can interact with a chip that is not already interacting with a terminal.

We formally describe the restricted model in Section 4 and prove the security of PACE v2 IM in this model.

Security in the BPR model. We analyze the security of PACE v2 IM in the so-called Bellare-Pointcheval-Rogaway (BPR) security model, and more precisely in the real-or-random setting. The real-or-random formulation of the BPR model is a refinement of BPR introduced by Abdalla, Fouque and Pointcheval [2]. This is a widely accepted model to capture the security requirements of a password-based key exchange protocol. In this security model, a number of users may engage in executions of the protocol with one another and the attacker is modeled as a probabilistic algorithm which initiates and controls these interactions. In the non-concurrent setting, users can only engage in one interaction at a time with a partner; when users can engage in several executions of the protocol with several partners simultaneously, the security model is said to be in the concurrent setting.

To prove the security of PACE v2 IM in the BPR model, we use a recent generic result from Fischlin *et al.* showing the security a large class of protocols belonging to the PACE framework [7]. Namely, the AKE

security of a protocol can be shown if a certain condition on the elliptic-curve mapping phase of the protocol is realized. Fischlin *et al.* captured this condition under the form of a security game gPACE-DH played between a challenger and an adversary. In Section 5 we therefore apply Fischlin *et al.* theorem by making explicit the gPACE-DH security game when applied to the mapping phase of PACE v2 IM; this enables to show that PACE v2 IM is secure in the BPR model.

2 Description of PACE v2 Integrated Mapping (IM)

We now describe PACE v2 IM, a password-based secure channel protocol proposed by Gemalto and Sagem Sécurité.

2.1 High-level features

- At the time when the protocol takes place, the chip and the terminal are assumed to share a common secret password π with low and given entropy. We will denote by N the size of the password dictionary *i.e.* the number of all possible passwords.
- The goal of PACE v2 IM is to make them generate a high-entropy ephemeral secret key material (one-time session keys and initialization vectors for secure messaging).
- Strong authentication means are not required at the chip or terminal side (no signing keys). The only input to PACE v2 IM is the password π.
- The protocol halts in three possible states:
  - secure channel established between chip and terminal (session keys generated on both sides);
  - the chip failed in password-authenticating the terminal (terminal failure);
  - the terminal failed in password-authenticating the chip (chip failure).
- PACE v2 IM is based on elliptic curves and is generic towards the choice of the elliptic curve. However it must be the case that $p = 2 \bmod 3$ where p is the characteristic of the prime field.
- PACE v2 IM relies on a list of cryptographic primitives and is generic towards the choice of these primitives. However, for PACE v2 IM to be securely implemented, it is important that the primitives be carefully chosen. At least one cryptographic suite shall be provided for secure implementations of PACE v2 IM.

2.2 Cryptographic primitives required in PACE v2 IM

The PACE v2 IM protocol makes use of

- an elliptic curve $\mathcal{E}/\mathbb{F}_p$ which is described as a set of domain parameters $\mathsf{params} = (p, a, b, G, q, f)$ where
  1. p is the characteristic of the prime field; its size $|p|$ typically ranges from 192 to 512 bits;
  2. a, b are the curve coefficients: the curve is defined as $\mathcal{E} = \{(x, y) \in \mathbb{F}_p^2 \mid y^2 = x^3 + ax + b \bmod p\}$;
  3. G is a generator of a subgroup $\mathcal{E}[q] \subseteq \mathcal{E}$ of prime order q;
  4. $\sharp\mathcal{E} = f \cdot q$ is the order of $\mathcal{E}$ where f is the co-factor;
- a Random Number Generator;
- a symmetric encryption scheme (block-cipher) $E : \{0,1\}^{\ell_0} \times \{0,1\}^{\ell_1} \mapsto \{0,1\}^{\ell_1}$ for some $\ell_0, \ell_1 \geq 0$. We refer to ℓ_0 as the key size of E and to ℓ_1 as its input and output size;
- a hash function $\mathcal{H}$ mapping arbitrary strings to $\{0,1\}^{|p|-1}$; its output is viewed as an integer in the interval $[0, 2^{p-1}[$; this integer can in turn be viewed as an element of $\mathbb{F}_p$, the set of integers modulo p;
- a function $\mathsf{Encoding} : \mathbb{F}_p \mapsto \mathcal{E}[q]$ which maps integers modulo p to points of order q on the elliptic curve; For practical security reasons, the so-called try and increment point encoding method [8] cannot be used as an embodiment of the $\mathsf{Encoding}$ function in PACE v2 IM (subsequent implementations suffer from partition attacks through execution time analysis). We therefore refer to Icart's point encoding technique [12] (described below) as the reference implementation for $\mathsf{Encoding}$ in PACE v2 IM.
- a key derivation function KDF mapping arbitrary strings to $\{0,1\}^{\ell}$ where $\ell = \ell_3 + \ell_4 + 2\ell_5 + 2\ell_6$;

– a secure messaging scheme combining a symmetric encryption scheme enc and a message authentication code mac as follows:
  - enc takes as inputs a symmetric key $k_{\mathsf{enc}} \in \{0,1\}^{\ell_3}$, possibly a state information $\mathsf{state}_{\mathsf{enc}} \in \{0,1\}^{\ell_4}$ (for instance a counter) and a plaintext $m \in \{0,1\}^*$ and outputs
$$(\mathsf{state}'_{\mathsf{enc}}, c) = \mathsf{enc}[k_{\mathsf{enc}}, \mathsf{state}_{\mathsf{enc}}](m)$$
  - enc^{-1} takes as inputs a symmetric key $k_{\mathsf{enc}} \in \{0,1\}^{\ell_3}$, possibly a state information $\mathsf{state}_{\mathsf{enc}} \in \{0,1\}^{\ell_4}$ (for instance a counter) and a ciphertext $c \in \{0,1\}^*$ and outputs:
$$(\mathsf{state}'_{\mathsf{enc}}, m) = \mathsf{enc}^{-1}[k_{\mathsf{enc}}, \mathsf{state}_{\mathsf{enc}}](c)$$
  We require that if $(\mathsf{state}'_{\mathsf{enc}}, c) = \mathsf{enc}[k_{\mathsf{enc}}, \mathsf{state}_{\mathsf{enc}}](m)$ then $(\mathsf{state}'_{\mathsf{enc}}, m) = \mathsf{enc}^{-1}[k_{\mathsf{enc}}, \mathsf{state}_{\mathsf{enc}}](c)$
  - mac takes as input a symmetric key $k_{\mathsf{mac}} \in \{0,1\}^{\ell_5}$, a state information $\mathsf{state}_{\mathsf{mac}} \in \{0,1\}^{\ell_6}$, a ciphertext $c \in \{0,1\}^*$ and outputs
$$(\mathsf{state}'_{\mathsf{mac}}, \mu) = \mathsf{mac}[k_{\mathsf{mac}}, \mathsf{state}_{\mathsf{mac}}](c)$$
  - **(encryption)** given the current secure messaging state information $(\mathsf{state}_{\mathsf{enc}}, \mathsf{state}_{\mathsf{mac}})$, computing the authenticated encryption of a plaintext $m \in \{0,1\}^*$ amounts to
    1. compute $(\mathsf{state}'_{\mathsf{enc}}, c) = \mathsf{enc}[k_{\mathsf{enc}}, \mathsf{state}_{\mathsf{enc}}](m)$
    2. compute $(\mathsf{state}'_{\mathsf{mac}}, \mu) = \mathsf{mac}[k_{\mathsf{mac}}, \mathsf{state}_{\mathsf{mac}}](c)$
    3. update $(\mathsf{state}_{\mathsf{enc}}, \mathsf{state}_{\mathsf{mac}}) = (\mathsf{state}'_{\mathsf{enc}}, \mathsf{state}'_{\mathsf{mac}})$
    4. return the authenticated ciphertext $c \,\|\, \mu$
  - **(decryption)** given the current secure messaging state information $(\mathsf{state}_{\mathsf{enc}}, \mathsf{state}_{\mathsf{mac}})$ and an authenticated ciphertext $c \,\|\, \mu$,
    1. compute $(\mathsf{state}'_{\mathsf{mac}}, \mu') = \mathsf{mac}[k_{\mathsf{mac}}, \mathsf{state}_{\mathsf{mac}}](c)$. If $\mu' \neq \mu$, reject the ciphertext as invalid.
    2. decrypt $(\mathsf{state}'_{\mathsf{enc}}, m) = \mathsf{enc}^{-1}[k_{\mathsf{enc}}, \mathsf{state}_{\mathsf{enc}}](c)$
    3. update $(\mathsf{state}_{\mathsf{enc}}, \mathsf{state}_{\mathsf{mac}}) = (\mathsf{state}'_{\mathsf{enc}}, \mathsf{state}'_{\mathsf{mac}})$
    4. return the plaintext m

  Typical implementations may use 3DES coupled with Retail MAC or AES 128/192/256 in counter mode coupled with CMAC.

The high-level view of PACE v2 IM given in this document remains generic towards the choice of these underlying cryptographic basic blocks and their size parameters $\ell_0, \ell_1, \ell_2, \ell_3, \ell_4, \ell_5, \ell_6 \geq 0$. Implementations for basic blocks (cryptographic suites) are not included in the current version of this document.

2.3 Description of PACE v2 IM

The protocol takes place between a chip (denoted A) and a terminal (denoted B).

Stage 0 (Parameter selection). A and B agree on the elliptic curve parameters
$$\mathsf{params} = (p, a, b, G, q, f)$$
and on the crypto suite
$$(E, \mathcal{H}, \mathsf{KDF}, \mathsf{enc}, \mathsf{mac}, \ell_0, \ldots, \ell_6)\ .$$
At this point, A and B are supposed to share a common password $\pi \in \{0,1\}^{\ell_0}$.

Stage 1 (Randomization).

1. A
   (a) randomly selects $s \leftarrow \{0,1\}^{\ell_1}$
   (b) sends $z = E_\pi(s)$ to B
2. B retrieves $s = E_\pi^{-1}(z)$

Notation: $E_k(m)$ denotes the encryption of a plaintext $m \in \{0,1\}^{\ell_1}$ under the key $k \in \{0,1\}^{\ell_0}$.

Stage 2 (Mapping).

1. B
   (a) randomly selects $\beta \leftarrow \{0,1\}^{\ell_2}$
   (b) sends β to A
2. they both compute $\hat{G} = \mathsf{Encoding}(\mathcal{H}(s \,\|\, \beta))$

Note that $s \,\|\, \beta$ is an $(\ell_1 + \ell_2)$-bit string, that $\mathcal{H}$ outputs $(|p| - 1)$-bit strings and that Icart's encoding function $\mathsf{Encoding}$ maps integers modulo p to points of $\mathcal{E}[q]$.

Stage 3 (Key establishment).

1. A
   (a) randomly selects $x \leftarrow [1, q-1]$
   (b) sends $X = x \cdot \hat{G}$ to B
2. B
   (a) randomly selects $y \leftarrow [1, q-1]$
   (b) sends $Y = y \cdot \hat{G}$ to A
3. they both compute $Z = xy \cdot \hat{G}$ and report a failure if $f \cdot Z = \mathcal{O}$

Stage 4 (Key derivation).

1. A and B individually derive the session key material

$$\mathsf{KDF}(Z \,\|\, X \,\|\, Y) = k_{\mathsf{enc}} \,\|\, \mathsf{state}_{\mathsf{enc}} \,\|\, k_{\mathsf{mac}} \,\|\, \mathsf{state}_{\mathsf{mac}} \,\|\, k'_{\mathsf{mac}} \,\|\, \mathsf{state}'_{\mathsf{mac}}$$

   and let $\mathsf{sk} = k_{\mathsf{enc}} \,\|\, \mathsf{state}_{\mathsf{enc}} \,\|\, k_{\mathsf{mac}} \,\|\, \mathsf{state}_{\mathsf{mac}}$.

Stage 5 (Key confirmation).

1. A computes $u = \mathsf{mac}(k'_{\mathsf{mac}}, (Y, \mathsf{params}))$ and sends u to B
2. B computes $v = \mathsf{mac}(k'_{\mathsf{mac}}, (X, \mathsf{params}))$ and sends v to A
3. each party checks the other's MAC and reports a failure in case of mismatch

Stage 6 (Session establishment). A and B initiate the security context $\mathsf{sk} = k_{\mathsf{enc}} \,\|\, \mathsf{state}_{\mathsf{enc}} \,\|\, k_{\mathsf{mac}} \,\|\, \mathsf{state}_{\mathsf{mac}}$ and use the secure messaging scheme to ensure encryption and integrity in further communications.

2.4 Icart's Encoding Function

Icart's encoding is a function that maps integers modulo p to points on any elliptic curve with coefficients a, b defined on the field of characteristic p (note that the function can be generalized to curves defined over field extensions $\mathbb{F}_{p^n}$ for $n > 1$) such that $p = 2 \bmod 3$. The point encoding is defined as

$$\begin{aligned} \mathsf{Encoding}: \quad [1, p-1] &\mapsto \mathbb{F}_p \times \mathbb{F}_p \\ u &\mapsto (x, y = ux + v) \end{aligned}$$

where

$$v = \frac{3a - u^4}{6u} \bmod p \qquad \text{and} \qquad x = \left(\left(v^2 - b - \frac{u^6}{27} \right)^{1/3} + \frac{u^2}{3} \right) \bmod p \,.$$

We then have that the point $(x, y) = \mathsf{Encoding}(u)$ belongs to the elliptic curve, as shown by Icart [12]. Note that

- $\mathsf{Encoding}(u)$ is not well-defined for $u = 0$. By convention, we set $\mathsf{Encoding}(0) = \mathcal{O}$ the neutral element of the elliptic curve,
- $\mathsf{Encoding}(u)$ defined as above does not necessarily yield a point of order q. Therefore when the cofactor f differs from 1 *i.e.* when $\mathcal{E}[q] \subsetneq \mathcal{E}$ then $\mathsf{Encoding}(u)$ returns the above point (x, y) multiplied by f.

Overall, $\mathsf{Encoding}(u)$ returns a point in the prime order subgroup $\mathcal{E}[q]$ for any input value u. An important property of Icart's point encoding is that it can be performed *in constant time*. Therefore computing the point encoding of a secret value can be achieved without leading to an information leakage on that secret value through timing analysis.

2.5 The Simplified SWU Algorithm

We recall the simplified Shallue-Woestijne-Ulas (SWU) algorithm described in [9]; the algorithm works for $p \equiv 3 \pmod 4$.

Simplified SWU algorithm:
Input: $\mathbb{F}_p$ such that $p \equiv 3 \pmod 4$, parameters a, b and input $t \in \mathbb{F}_p$
Output: $(x, y) \in E_{a,b}(\mathbb{F}_p)$ where $E_{a,b} : y^2 = x^3 + ax + b$

1. $\alpha \leftarrow -t^2$
2. $X_2 \leftarrow \frac{-b}{a}\left(1 + \frac{1}{\alpha^2+\alpha}\right)$
3. $X_3 \leftarrow \alpha \cdot X_2$
4. $h_2 \leftarrow (X_2)^3 + a \cdot X_2 + b$; $h_3 \leftarrow (X_3)^3 + a \cdot X_3 + b$
5. If h_2 is a square, return $(X_2, h_2^{(q+1)/4})$, otherwise return $(X_3, h_3^{(q+1)/4})$

As showns in [9] this defines a point $(x, y) = \mathsf{Encoding}(t)$ that belongs to the elliptic curve. Note that as for Icart's function $\mathsf{Encoding}(u)$ defined as above does not necessarily yield a point of order q. Therefore when the cofactor f differs from 1 *i.e.* when $\mathcal{E}[q] \subsetneq \mathcal{E}$ then $\mathsf{Encoding}(u)$ must return the above point (x, y) multiplied by f.

2.6 Differences with the German PACE v2 Generic Mapping (GM)

BSI [10] has defined a password-based secure channel protocol called PACE v2 GM ($\neq$ PACE v2 IM). The two protocols are very similar in nature and differ only in Stage 2 (the so-called mapping phase). In PACE v2 GM, the mapping phase is defined as follows.

Stage 2 (Mapping) in the German PACE v2 GM.

1. A
   (a) randomly selects $\alpha \leftarrow [1, q-1]$
   (b) sends $\alpha \cdot G$ to B
2. B
   (a) randomly selects $\beta \leftarrow [1, q-1]$
   (b) sends $\beta \cdot G$ to A
3. they both compute $H = (\alpha\beta) \cdot G$ and report a failure if $f \cdot H = \mathcal{O}$
4. they both compute $\hat{G} = H + s \cdot G$

All the other stages are strictly identical between PACE v2 IM and PACE v2 GM. Protocols that follow the same methodology, namely where only the mapping phase is specific, are said to belong to the *PACE framework*. Hence both the German PACE v2 GM and PACE v2 IM belong to the PACE framework and can be implemented at the APDU level with the same set of commands.

It is important to note that PACE v2 IM requires the two primitives $\mathcal{H}$ and $\mathsf{Encoding}$ to be included in the crypto suites that support the protocol and that the German PACE v2 GM does not require them. Also note that PACE v2 IM enjoys increased performances as compared with PACE v2 GM: PACE v2 GM requires 5 scalar multiplications at the chip and terminal side whereas PACE v2 IM requires only 2. Elliptic curve operations constitute the main part of the execution time of both protocols since the other primitives are negligible in comparison.

Since it is desired for a password-based secure channel protocol to realize forward security (see below), some form of Diffie-Hellman key agreement is required at some point during the protocol. Therefore at least 2 scalar multiplications must be performed both at the chip and terminal side. PACE v2 IM provides password authentication on top of this, at negligible extra cost.

2.7 A note on intellectual property rights

A number of patents exist which protect cryptographic protocols and techniques dedicated to password-authenticated key establishment. We refer in particular to the EKE [5,6] and SPEKE [13] protocol designs. PACE v2 IM has been designed, among a number of possible alternatives, in order to evade all known intellectual property claims. In our quest for a totally patent-free mechanism, we have attempted to take all preexisting patents into consideration to ensure that PACE v2 IM remains independent from their scope, and we believe that we have reached a high level of certainty in this respect. We emphasize, however, that no formal guarantee can be given that no absolutely third party in the world owns IP rights on subparts or characteristics of the design. Also, we repeat that PACE v2 IM makes use of a point encoding function and that this technique is patent-pending and owned by Sagem Sécurité. Sagem Sécurité has agreed to grant free-of-charge exploitation rights under restrictive conditions [1]. This patent is the only known IP right holding on PACE v2 IM.

3 Security Models for Password-Based Secure Channels

Since there is a rather limited number of passwords, it may be possible for an attacker to enumerate, in an offline way, all possible passwords from the (public) password dictionary; this is called a dictionary attack. We would like that the adversary's ability in learning the session key sk only depends on her/his online interaction with the chip or the terminal, and that it doesn't significantly depend on her/his off-line computing time or power. In other words, the goal of a password-based key exchange protocol is to make sure that the adversary cannot be more successful than by simply randomly guessing the password during the online phase where access is given to the honest parties. The system can then limit the number of interactions so that a password can be invalidated after a certain number of failed attempts.

3.1 The Passive Security Model (Eavesdropping)

Eavesdropping is the weakest form of an adversarial scenario. In this (very restricted) security model, the attacker is assumed to listen to messages exchanged between honest parties without being able to impersonate them or modify the contents of the transmissions. Passive attackers capture the threat of real-life attack scenarios where *e.g.* an antenna is actively recording transactions and intensive computational power takes place afterward to recover passwords and/or session keys.

Passive attacks may have several goals. We distinguish between the following security breaches:

- password recovery: this is also known as off-line dictionary attacks. Namely, the attacker collects a number of protocol traces and extracts the password(s) from these using computational power. Given that the password dictionary has limited entropy in typical applications, this type of attack is often considered as a major risk;
- secrecy: the attacker aims at recovering one or several session keys given the monitored traces. This is also known as *channel opening*;
- forward secrecy: the attacker knows the password and attempts to open monitored channels.

It is desired that a password-based protocol resists these 3 types of attacks. However the passive security model is often considered as insufficient to capture real-life threats where attackers may well engage in a transaction with a device without having the holder noticing anything. As a result, security requirements must include, but should not be limited to, purely passive attacks.

3.2 The Chip-only or Terminal-only Active Security Model

A refinement of passive adversaries consists in considering scenarios where a target device is made available to the attacker. Attacks may then combine pure eavesdropping and active dialog with the device, be it a chip or a terminal. In this security model, the attacker is assumed to interact with only one device and attempts

to break either the password (password recovery) or a session key (secrecy). Forward secrecy may also be defined in this context; however it is obvious that forward secrecy cannot be realized (*i.e.* by any protocol) if the attacker is granted active access to the device *after* the password is revealed. However forward secrecy makes sense when no access to the target device is possible (but eavesdropping is allowed) after the password is disclosed to the attacker.

This security model encompasses much more realistic attacks that the pure passive model of attackers. In particular, it captures the threat of relay and skimming attacks. However yet stronger attacks can be taken into account.

3.3 The Bellare-Pointcheval-Rogaway Security Model

The most general and powerful security model for password-based key exchange has been defined by Bellare, Pointcheval and Rogaway in [4]. In this model, one considers a set of clients $C \in \mathcal{C}$ and a set of servers $S \in \mathcal{S}$; a client C and a server S communicate via a client instance C_i and a server instance S_j. In the concurrent model, the adversary can create several instances of a participant that can be simultaneously active, whereas in the non-concurrent model only one active user instance U_i is allowed between user U and server S. The adversary is granted access to plenty of honest executions; namely only passive eavesdropping is required to access them. Moreover the adversary can request session keys between some user instance U_i and server instance S_j; the loss of a session key should not compromise the security of other sessions. Eventually the adversary tries to tell apart a real session key used by U_i and S_j (not revealed) from a random one. In this model the adversary can also corrupt users to obtain their passwords; then previously established session-keys should remain secure; this captures the notion of forward secrecy in this context.

It is widely agreed that the BPR (usually non-concurrent) security model is strong enough to capture all practical threats a device may have to face in practice. The concurrent setting gives yet another refinement in the case where devices support several threads.

3.4 The Random Oracle and Ideal Cipher Models

In this document, security is proved using the Random Oracle (RO) and the Ideal Cipher (IC) models. In the random oracle model, one assumes that some hash function is replaced by a publicly accessible random function (the random oracle). This means that the adversary cannot compute the result of the hash function by herself: she must query the random oracle. Similarly in the ideal cipher model, a block cipher (*i.e.* a permutation parametrized by a key) is modeled as a permutation oracle. Although it is better to obtain a key-exchange protocol not relying on the RO or IC models (such as [5]), these models are a commonly used formal tool for evaluating the security level of efficient constructions.

4 A Security Analysis of PACE v2 IM in the Restricted Model

4.1 The Restricted Model

In this section we consider the following simplifications to the Bellare-Pointcheval-Rogaway security model [4]:

1. The chip cannot interact with more than one terminal at the same time, and the terminal cannot interact with more than one chip at the same time (non-concurrent model).
2. The terminal does not hold any secret, except the chip's password.
3. The adversary cannot tamper with the messages sent by the chip and the terminal during their interaction; it can only eavesdrop their communications. In particular, the adversary cannot interact with a terminal holding a password; however, the adversary can interact with a chip that is not already interacting with a terminal.

More precisely, we consider the following game between an attacker and a challenger. The adversary can interact at most k times with the chip, without knowing the password; at the end of each interaction, the adversary can request the corresponding session key. Eventually the adversary can request the password (forward secrecy), and he must output a session key that was not previously revealed.

1. The challenger generates the system public parameters params and sends params to the attacker.
2. The challenger generates a password π at random in the dictionary $\mathcal{D}$.
3. Using π, the challenger simulates n protocols executions between a chip and a terminal, and sends the transcripts $(T_1, \ldots, T_n)$ to the attacker.
4. The attacker can request any of the session keys used in the transcripts T_i.
5. The attacker can engage in up to k protocol executions with the challenger. The challenger answers using π. At the end of each execution, the attacker can request the corresponding session key sk_i.
6. The challenger sends password pw to the attacker (forward security).
7. The attacker must output one of the session keys, not revealed previously.

For each of the k interactions with the chip, the attacker can succeed with probability $1/|\mathcal{D}|$, by simply guessing a password at random in the dictionary $\mathcal{D}$. Then for k interactions, the attacker's success probability can be at least $k/|\mathcal{D}|$. We say that a password-based key exchange is secure if the adversary's success probability cannot be significantly larger:

Definition 1. *A password-based key exchange protocol with passwords in dictionary $\mathcal{D}$ is said to be $(n, k, \tau, \varepsilon)$-secure if no adversary can recover a session key with probability greater than $k/|\mathcal{D}| + \varepsilon$, while getting at most n protocol transcripts and running in time less than τ.*

4.2 Computational Assumptions

The security of PACE v2 IM relies on a computational assumption which we call the *Gap Chosen-Base Diffie-Hellman (GCBDH) assumption.*

Definition 2 (DDH Oracle). *Recall that $\mathcal{E}[q]$ is the q-torsion subgroup of the elliptic curve $\mathcal{E}$. A DDH oracle on $\mathcal{E}[q]$ is an oracle which, given any tuple $(G, uG, vG, wG) \in \mathcal{E}[q]^3$, outputs 1 when $w = uv \bmod q$ and 0 otherwise. Here we consider perfect DDH oracles i.e. that have an error probability equal to zero.*

Definition 3 (The CBDH problem). *Solving the CBDH problem on $\mathcal{E}[q]$ consists, on input a random tuple $(G, aG, bG) \leftarrow \mathcal{E}[q]^3$, in finding a tuple $(H, \frac{1}{a}H, \frac{1}{b}H) \in \mathcal{E}[q]^3$ with $H \neq \mathcal{O}$.*

Definition 4 (The GCBDH problem). *Solve CBDH on $\mathcal{E}[q]$ given a DDH oracle on $\mathcal{E}[q]$.*

The GCBDH problem is said to be $(q_{\mathrm{DDH}}, \tau, \varepsilon)$-intractable if no probabilistic algorithm running in time at most τ can solve a random instance of GCBDH with probability at least ε, in at most q_{DDH} calls to the DDH oracle. It is easily seen that GCBDH is not harder than the computational Diffie-Hellman (CDH) problem on $\mathcal{E}[q]$; namely an algorithm solving CDH can be used to solve GCBDH. However the converse is not known to be true.

4.3 Security Result on PACE v2 IM

We now show that PACE v2 IM is secure with respect to the security notion as per Definition 1 but without considering forward secrecy (Step 6 in the scenario of Definition 1). This security notion encompasses all the different flavors of session key secrecy, resistance against offline dictionary attacks, and so forth, except forward secrecy, which will be considered in the BPR model (see Section 5).

The security game of Definition 1, adapted to the case of PACE v2 IM, gives the following attack scenario played by the attacker $\mathcal{A}$ and a challenger $\mathcal{C}$. For simplicity we denote by $\mathsf{map2point} : \{0,1\}^{\ell_1+\ell_2} \to \mathcal{E}[q]$ the function:

$$\mathsf{map2point}(s, \beta) := \mathsf{Encoding}(\mathcal{H}(s \,\|\, \beta))$$

1. The challenger $\mathcal{C}$ generates params and sends params to the attacker.
2. $\mathcal{C}$ selects a password $\pi^* \leftarrow \mathcal{D}$ uniformly at random
3. *Offline phase:* Using π^*, $\mathcal{C}$ generates n protocol executions between a chip and a terminal and sends the transcripts $T_1, \ldots, T_n$ to the adversary $\mathcal{A}$.
4. The adversary $\mathcal{A}$ can request any of the session keys $\mathsf{sk}_1, \ldots, \mathsf{sk}_n$ matching $(T_1, \ldots, T_n)$
5. *Online phase:* The following is repeated k times:
   (a) $\mathcal{C}$ selects $s \leftarrow \{0,1\}^{\ell_1}$ and sends $z = E_{\pi^*}(s)$ to $\mathcal{A}$
   (b) $\mathcal{A}$ sends $\beta \in \{0,1\}^{\ell_2}$ to $\mathcal{C}$
   (c) $\mathcal{C}$ generates $\hat{G} = \mathsf{map2point}(s, \beta)$
   (d) $\mathcal{C}$ selects $x \leftarrow \mathbb{Z}_q$ and sends $X = x\hat{G}$ to $\mathcal{A}$
   (e) $\mathcal{A}$ sends a point Y to $\mathcal{C}$ (wlog $Y \in \mathcal{E}[q] \setminus \{\mathcal{O}\}$)
   (f) $\mathcal{C}$ computes $Z = xY$ and $\mathsf{KDF}(Z \,\|\, X \,\|\, Y) = \mathsf{sk} \,\|\, k'_{\mathsf{mac}} \,\|\, \mathsf{state}'_{\mathsf{mac}}$
   (g) $\mathcal{A}$ may request the session key sk; in this case, $\mathcal{C}$ sends sk to $\mathcal{A}$
6. Eventually the adversary $\mathcal{A}$ outputs one of the session keys sk, not revealed by the challenger $\mathcal{C}$.

Theorem 1. *Assume that G-CBDH is (τ, ε)-hard on $\mathcal{E}[q]$. Then PACE v2 IM is $(n, k, \tau', \varepsilon')$-secure where*

$$\varepsilon' = \varepsilon - \frac{(n+k)^2}{2^{\ell_1}} - \frac{q_h^2}{2^{\ell_1}} - 2^{-\ell_{\mathsf{sk}}} - 3 \cdot \varepsilon_{GCBDH} - q_h \cdot 2^{-\lambda} \tag{1}$$

$$\tau' = \tau - O(q_h(n+k)) \tag{2}$$

where q_h is the number of oracle queries made by the adversary and λ is a security parameter.

4.4 Proof of Theorem 1

We consider an attacker $\mathcal{A}$ against PACE v2 IM which follows the security game described above, and show how to construct a reduction algorithm $\mathcal{R}$ which solves the GCBDH problem over $\mathcal{E}[q]$. We view the hash function KDF as a random oracle and the block-cipher E as an ideal cipher. Additionally we first view the hash function $\mathsf{map2point} = \mathsf{Encoding} \circ \mathcal{H}$ as a random oracle; we then show how to adapt the proof when only $\mathcal{H}$ is viewed as a random oracle. We use a sequence of games; we denote by S_i the event that the attacker succeeds in $\mathtt{Game}_i$. We recall the Difference Lemma [15]:

Lemma 1 (Difference Lemma). *Let A, B, F be events defined in some probability distribution, and suppose that $A \wedge \neg F \Leftrightarrow B \wedge \neg F$. Then $|\Pr[A] - \Pr[B]| \leq \Pr[F]$.*

$\mathtt{Game}_0$: this is the attack scenario.

$\mathtt{Game}_1$: we modify the way the s values are generated, both in the offline and in the online phase. Namely we generate the s values in such a way that there are all distinct. Therefore $\mathtt{Game}_0$ and $\mathtt{Game}_1$ are the same unless there is a collision among the s values, which happens with probability at most $(n+k)^2/2^{\ell_1}$; this gives:

$$|\Pr[S_1] - \Pr[S_0]| \leq \frac{(n+k)^2}{2^{\ell_1}} \tag{3}$$

$\mathtt{Game}_2$: we consider the offline phase. In every offline transcript instead of generating the secret-key sk as $\mathsf{KDF}(Z \,\|\, X \,\|\, Y) = \mathsf{sk} \,\|\, k'_{\mathsf{mac}} \,\|\, \mathsf{state}'_{\mathsf{mac}}$ we simply generate sk at random in $\{0,1\}^{\ell_{\mathsf{sk}}}$, where $\ell_{\mathsf{sk}} = \ell_3 + \ell_4 + \ell_5 + \ell_6$.

Let F be the event that the adversary makes a KDF query to $Z \,\|\, X \,\|\, Y$ in Game 2 for some $Z \,\|\, X \,\|\, Y$ in one of the offline transcripts. It is clear that Games 1 and 2 proceed identically unless F occurs; therefore by the Difference Lemma:

$$|\Pr[S_2] - \Pr[S_1]| \leq \Pr[F]$$

We claim that

$$\Pr[F] \leq \varepsilon_{GCBDH}$$

where ε_{GCBDH} is the probability of solving the GCBDH problem for some efficient algorithm C, which gives:

$$|\Pr[S_2] - \Pr[S_1]| \leq \varepsilon_{GCBDH} \tag{4}$$

Algorithm C runs as follows: it receives as input (G, aG, bG) and must output $(H, \frac{1}{a}H, \frac{1}{b}H)$ for some H; algorithm C interacts with the adversary, playing the role of the challenger in Game 2. However when the adversary or the challenger makes a query $\mathsf{map2point}(s, \beta)$ for some (s, β) appearing in the offline phase, algorithm C returns $\hat{G} = uG$ for some random $u \in \mathbb{Z}_q$ (instead of a random point). This is always possible since in Game 2 the elements s from the offline (and online) phase are all distinct. Moreover in the offline phase transcripts instead of returning $X = x\hat{G}$ and $Y = y\hat{G}$ it returns $X = v(aG)$ and $Y = w(bG)$ for some random $v, w \in \mathbb{Z}_q$. Clearly this does not modify the distribution of the random variables appearing in Game 2.

Then for every query $\mathsf{KDF}(Z' \,\|\, X \,\|\, Y)$ where (z, β, X, Y) appears in one of the offline transcripts, algorithm C calls the DDH oracle to determine whether $(\hat{G}, X, Y, Z')$ is a DDH 4-uple, where $\hat{G} = \mathsf{map2point}(s, \beta)$ where $s = E_\pi^{-1}(z)$. In this case this means that event F has occurred in Game 2. Since $X = v(aG) = (va/u)\hat{G}$ and $Y = w(bG) = (wb/u)\hat{G}$ this gives:

$$Z' = \frac{va}{u}\frac{wb}{u}\hat{G} = \frac{vwab}{u}G$$

from which one can compute abG. This enables to output (abG, bG, aG), thereby solving the GCBDH problem.

$\mathtt{Game}_3$: we modify the way the s values are generated in both the offline and online phase. Instead of letting $s \leftarrow \{0,1\}^{\ell_1}$ and then $z = E_\pi(s)$, we first let $z \leftarrow \{0,1\}^{\ell_1}$ and then $s = E_\pi^{-1}(z)$. Moreover, as in Games 1 and 2 the z values in both the offline and online phases are generated in such a way that they are all distinct. Clearly the distribution of the s and z random variables is the same in Games 2 and 3; therefore:

$$\Pr[S_3] = \Pr[S_2] \tag{5}$$

$\mathtt{Game}_4$: we abort if for any z_i, z_j appearing in the offline or online phase with $i \neq j$, we have $E_a^{-1}(z_i) = E_b^{-1}(z_j)$ for some a, b appearing in some E or E^{-1} query. Since there are at most q_h such queries, we have:

$$|\Pr[S_4] - \Pr[S_3]| \leq \frac{q_h^2}{2^{\ell_1}} \tag{6}$$

$\mathtt{Game}_5$: we proceed as in Game 2 for the online phase: in each of the k step of the online phase we simply generate sk at random in $\{0,1\}^{\ell_{\mathsf{sk}}}$, instead of letting sk from $\mathsf{KDF}(Z \,\|\, X \,\|\, Y) = \mathsf{sk} \,\|\, k'_{\mathsf{mac}} \,\|\, \mathsf{state}'_{\mathsf{mac}}$.

It is clear that in Game 5 the adversary's view is independent of all the session keys; therefore its success probability is no better than random guessing:

$$\Pr[S_5] \leq 2^{-\ell_{\mathsf{sk}}} \tag{7}$$

We also argue that in Game 5 the adversary's view is also independent of the password π. Namely the offline and online transcript consist in (z, β, X, Y) (and optionally sk), and in Game 5 all these values are generated independently of the password π.

As in Game 2 we denote by F the event that the adversary makes a KDF query to $Z \,\|\, X \,\|\, Y$ in Game 5 for some $Z \,\|\, X \,\|\, Y$ in one of the online transcripts. It is clear that Games 4 and 5 proceed identically unless F occurs; therefore by the Difference Lemma:

$$|\Pr[S_4] - \Pr[S_5]| \leq \Pr[F]$$

As in Game 2 we claim that

$$\Pr[F] \leq \frac{k}{|\mathcal{D}|} + 2 \cdot \varepsilon_{GCBDH} \tag{8}$$

where ε_{GCBDH} is the probability of solving the GCBDH problem for some efficient algorithm C. This gives:

$$|\Pr[S_4] - \Pr[S_5]| \leq \frac{k}{|\mathcal{D}|} + 2 \cdot \varepsilon_{GCBDH} \tag{9}$$

Given online transcript (z, β, X, Y) we denote by $S(z, \beta, X, Y)$ the set of $r \in \mathcal{D}$ such that the adversary has made a KDF query to $Z' \,\|\, X \,\|\, Y$ and (G', X, Y, Z') is a DDH 4-uple, where $G' = \mathsf{map2point}(s, \beta)$ with $s = E_r^{-1}(z)$. We denote by Q_1 the event that for all online transcripts (z, β, X, Y) the size of $S(z, \beta, X, Y)$ is either 0 or 1; then $Q_2 = \neg Q_1$ is the event that for at least one online transcript (z, β, X, Y) the size of $S(z, \beta, X, Y)$ is at least 2.

Assume that event Q_1 occurs and let (z, β, X, Y) be an online transcript such that the size of $S(z, \beta, X, Y)$ is 1 (if any). Let $r \in \mathcal{D}$ such that the adversary has made a KDF query to $Z' \,\|\, X \,\|\, Y$ and (G', X, Y, Z') is a DDH 4-uple, where $G' = \mathsf{map2point}(s, \beta)$ with $s = E_r^{-1}(z)$. Since from Game 4 all $E_{r'}^{-1}(z)$ are distinct we have that event F can only occur if $\pi = r$; since in Game 5 the distribution of π is independent from the adversary's view this happens with probability $1/|\mathcal{D}|$; since there are at most k online transcripts event F occurs with probability at most $k/|\mathcal{D}|$, which gives:

$$\Pr[F|Q_1] \leq \frac{k}{|\mathcal{D}|}$$

Therefore we have:

$$\Pr[F] = \Pr[F|Q_1] \cdot \Pr[Q_1] + \Pr[F|Q_2] \cdot \Pr[Q_2] \leq \frac{k}{|\mathcal{D}|} + \Pr[Q_2]$$

Therefore to prove inequality (8) it suffices to show that $\Pr[Q_2] \leq 2 \cdot \varepsilon_{GCBDH}$. To prove this latter inequality we describe an algorithm C that solves the GCBDH problem when interacting with the adversary; algorithm C plays the role of the challenger in Game 5.

Algorithm C runs as follows: it receives as input (G, aG, bG) and must output $(H, \frac{1}{a}H, \frac{1}{b}H)$ for some H. When the adversary or the challenger makes a fresh query $\mathsf{map2point}(s, \beta)$ where (s, β) is not from the offline phase, algorithm C flips a coin and returns either $\hat{G} = u(aG)$ or $\hat{G} = u(bG)$ with equal probability, for some random $u \in \mathbb{Z}_q$ (instead of a random point). Moreover in the online phase instead of returning $X = x\hat{G}$ it returns $X = cG$ for some random $c \in \mathbb{Z}_q$. Clearly this does not modify the distribution of the random variables appearing in Game 5.

Assume that event Q_2 occurs, and let (z, β, X, Y) be an online transcript such that the size of $S(z, \beta, X, Y)$ is at least 2. Let $r_1, r_2 \in S(z, \beta, X, Y)$ and let $s_1 = E_{r_1}^{-1}(z)$ and $s_2 = E_{r_2}^{-1}(z)$. Let $\hat{G}_1 = \mathsf{map2point}(s_1, \beta)$ and $\hat{G}_2 = \mathsf{map2point}(s_2, \beta)$. The adversary has made two KDF queries $Z_1 \,\|\, X \,\|\, Y$ and $Z_2 \,\|\, X \,\|\, Y$ such that $(\hat{G}_1, X, Y, Z_1)$ and $(\hat{G}_2, X, Y, Z_2)$ are both DDH 4-uples. Algorithm C can detect these queries using the DDH oracle. Now with probability $1/2$ we have $\hat{G}_1 = u(aG)$ and $\hat{G}_2 = v(bG)$ (or the opposite) for some $u, v \in \mathbb{Z}_q$. Using:

$$X = cG = \frac{c}{ua}\hat{G}_1 = \frac{c}{vb}\hat{G}_2$$

this gives:

$$Z_1 = \frac{c}{ua}Y, \quad Z_2 = \frac{c}{vb}Y$$

Therefore algorithm C can let $H := Y$ and compute $\frac{1}{a}H = \frac{u}{c}Z_1$ and $\frac{1}{b}H = \frac{v}{c}Z_2$, thereby solving the GCBDH problem with probability $1/2$. Therefore we must have:

$$\Pr[Q_2] \leq 2 \cdot \varepsilon_{GCBDH}$$

Combining inequalities (3), (4), (5), (6), (7) and (9), we obtain:

$$\Pr[S_0] \leq \frac{k}{|\mathcal{D}|} + \frac{(n+k)^2}{2^{\ell_1}} + \frac{q_h^2}{2^{\ell_1}} + 2^{-\ell_{\mathsf{sk}}} + 3 \cdot \varepsilon_{GCBH}$$

In the arguments above we have viewed the function map2point as a random oracle into the curve. However this is not a reasonable model since map2point = Encoding $\circ\, \mathcal{H}$ and the Encoding function only generates a fraction of the elliptic-curve points.

The following Lemma, whose proof is given in Appendix A, shows that Encoding is simulatable *i.e.* fulfills the following property. We stress that the lemma holds for both Icart's function and the simplified SWU algorithm recalled in sections 2.4 and 2.5.

Lemma 2. *Let $\lambda \geq 0$ be a security parameter. There exists a probabilistic polynomial time algorithm $\mathcal{S}$ which on input an element $G \in \mathcal{E}[q]$ outputs a pair $(\alpha, \rho) \in (\mathbb{F}_p \cup \{\perp\}) \times \mathbb{F}_q$ such that* $\mathsf{Encoding}(\alpha) = \rho G$ *when $\alpha \neq \perp$, and the the distribution of α is $2^{-\lambda}$-statistically close to the uniform distribution over $\mathbb{F}_p$.*

For simplicity we first we view $\mathcal{H}$ as a hash oracle into $\mathbb{F}_p$ (instead of $\{0,1\}^{|p|-1}$). In the previous proof where map2point is seen as a random oracle, algorithm C for the offline phase simulates the answer of a $\mathsf{map2point}(s, \beta)$ query by returning $\hat{G} = uG$ for some random $u \in \mathbb{Z}_q$. Instead when only $\mathcal{H}$ is seen as a random oracle into $\mathbb{F}_p$, algorithm C will simulate the answer of a $\mathcal{H}(s, \beta)$ query by running the previous algorithm $\mathcal{S}$ with input G, obtaining (α, ρ) and letting $\mathcal{H}(s, \beta) = \alpha$, which gives $\mathsf{map2point}(s, \beta) = \mathsf{Encoding}(\mathcal{H}(s, \beta)) = \mathsf{Encoding}(\alpha) = \rho G$. Then the previous proof remains the same with $u := \rho$. Similarly for the online phase algorithm C uses aG or bG (with equal probability) as input of algorithm $\mathcal{S}$. Since from Lemma 2 the distribution of u is $2^{-\lambda}$-statistically close to the uniform distribution in $\mathbb{F}_p$ this only adds a (negligible) term $q_h \cdot 2^{-\lambda}$ in the security bound, where q_h is the number of hash queries.

Finally, when $\mathcal{H}$ is seen as a hash oracle into $[0, 2^{|p|-1}[$ (instead of $\mathbb{F}_p$), it is easy to modify the previous algorithm $\mathcal{S}$ so that it outputs $\alpha \in [0, 2^{|p|-1}[$ instead of $\mathbb{F}_p$. For this one simply runs $\mathcal{S}$ until it outputs $\alpha \in [0, 2^{|p|-1}[$. One gets a new probabilistic algorithm $\mathcal{S}'$ which remains polynomial-time; moreover the distribution of α can still be made $2^{-\lambda}$-statistically close to the uniform distribution over $[0, 2^{|p|-1}[$. This terminates the proof of Theorem 1.

5 Security Proof for PACE v2 IM in the BPR Model

This section considers the security of PACE v2 IM in the Bellare-Pointcheval-Rogaway (BPR) security model [4], and more precisely in the real-or-random setting. The real-or-random formulation of the BPR model is a refinement of BPR introduced by Abdalla, Fouque and Pointcheval [2]. In this security model, a number of users may engage in executions of the protocol with one another and the attacker is modeled as a probabilistic algorithm which initiates and controls these interactions. In the non-concurrent setting, users can only engage in one interaction at a time with a partner; when users can engage in several executions of the protocol with several partners simultaneously, the security model is said to be in the concurrent setting. In what follows, we focus on the standard, non-concurrent setting.

The BPR formalization allows to capture all the possible goals an attacker may pursue when attempting to break the protocol: off-line dictionary attacks (*i.e.* recovering the password), secrecy (opening a secure channel) and forward secrecy (opening past secure channels given the password) are all covered by specifying which type of attack path the adversary is allowed to follow. Here an attack path means a sequence of interactions with users (which the attacker views as oracles and to which she can make oracles calls) with specific constraints intended to avoid trivial breaks that would make the security game pointless. For instance, if the attacker is allowed to force a user to reveal its password, then secure channels created by that user in future protocol runs where the attacker plays the role of the partner can obviously be opened. Therefore, in order to capture forward secrecy properly, the adversary is requested to proceed by other means. We refer to the papers given in reference for more detail [2,4].

In the real-or-random BPR model, one can define a generic security notion which covers all the desired security properties of the protocol. This notion is referred to as AKE security and formalizes the intuition that the attacker cannot tell the difference between a genuine execution of the protocol and a fake protocol execution where the created session key is replaced with a random value. Clearly if we can show that no adversary, even when interacting with users (chips and terminals alike) can tell apart correct session keys from random session keys, this gives the strongest possible evidence that the protocol can be safely used in

practice. We define the AKE advantage of an adversary $\mathcal{A}$ for a key agreement protocol P by (see [2] for the details):

$$\mathsf{Adv}_P^{ake}(\mathcal{A}) := 2 \cdot \Pr[\mathcal{A} \text{ wins}] - 1$$
$$\mathsf{Adv}_P^{ake}(t, Q) := \max\left\{\mathsf{Adv}_P^{ake}(\mathcal{A}) \mid \mathcal{A} \text{ is } (t, Q)\text{-bounded}\right\}$$

where t is the running time of the adversary and $Q = (q_e, q_h, q_c)$, where q_e is the number of initiated executions, q_h the number of hash queries, and q_c the number of cipher queries.

5.1 Fischlin *et al.* Generic Theorem

Fischlin *et al.* recently came up with a nice generic result that allows to show the security of protocols belonging to the PACE framework [7]. Namely, the AKE security of a protocol can be shown if a certain condition on the mapping phase is realized. Fischlin *et al.* captured this condition under the form of a security game gPACE-DH played between a challenger and an adversary. The description of gPACE-DH depends on the mapping phase and thus can be very different from one protocol to another in the PACE framework. Fischlin *et al.* generic theorem [7] states that if the gPACE-DH security game is shown to be intractable, then the whole protocol will be proved AKE secure. In the following we therefore apply Fischlin *et al.* theorem by making explicit the gPACE-DH security game when applied to the mapping phase of PACE v2 IM.

Definition 5 (gPACE-DH Problem [7], Def. 4.3). *The general password-based chosen-element DH problem is $(t, N, q_\ell, \varepsilon)$-hard (with respect to* map2point*) if for any adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$ running in total time t the probability that the following experiment returns 1 is at most $1/N + \varepsilon$:*

1. *Pick* params*, including a generator G.*
2. *Let $(st_0, s_1, \ldots, s_N) \leftarrow \mathcal{A}_0(\mathsf{params}, N)$*
3. *Pick $y_B \leftarrow \mathbb{Z}_q$ and $k \leftarrow [1, N]$*
4. *Let $\hat{G}$ be the local output of the honest party in an execution of* map2point(s_k)*, where $\mathcal{A}_1(st_0)$ controls the other party (and generates the local state st_1).*
5. *let $(Y_A, K_1, \ldots, K_{q_\ell}) \leftarrow \mathcal{A}_2(st_1, y_B\hat{G})$*
6. *output 1 iff $Y_A \neq 0$ and $K_i = y_B Y_A$ for some i*

One lets $\mathsf{Adv}_{\mathsf{map2point}}^{gPACE-DH}(t, N, q_\ell)$ denote a value ε for which the gPACE-DH problem is $(t, N, q_\ell, \varepsilon)$-hard (with respect to map2point).

Theorem 2 (Fischlin *et al.* [7], Th 5.1). *Let* map2point *be canonical and assume that the password is chosen from a dictionary of size N. In the random oracle model and in the ideal cipher model we have:*

$$\mathsf{Adv}_{PACE}^{ake}(t, Q) \leq \frac{q_e}{N} + q_e \cdot \mathsf{Adv}_{\mathsf{map2point}}^{gPACE-DH}(t^*, N, q_h) + q_e \cdot \varepsilon_{\text{forge}} + \frac{2q_eN^2 + 9q_e^2N + q_cq_e}{\min\{q, |Range(\mathcal{H})|\}}$$

where $t^* = t + \mathcal{O}(kq_e^2 + kq_h^2 + kq_c^2 + k^2)$ and $Q = (q_e, q_c, q_h)$, where k is a security parameter, q_e is the number of protocol executions launched by the adverary, q_h and q_e indicate the number of hash and cipher queries, and $\varepsilon_{\text{forge}}$ is the probability of forging a MAC in less than $2q_e$ adaptive MAC queries and time less than t^*.

5.2 Application to PACE v2 IM

In the following we apply Fischlin *et al.* Theorem 2 by making explicit the gPACE-DH security game when applied to the mapping phase of PACE v2 IM. We obtain a security game G1 played between the attacker $\mathcal{A}$ and a challenger $\mathcal{C}$.

Security Game G1:

1. $\mathcal{C}$ generates public parameters and sends them to $\mathcal{A}$
2. $\mathcal{A}$ arbitrarily chooses N pairwise distinct values $s_1, \ldots, s_N \in \{0,1\}^{\ell_1}$ and sends them to $\mathcal{C}$
3. $\mathcal{C}$:
   (a) randomly selects $\beta \leftarrow \{0,1\}^{\ell_2}$
   (b) randomly selects $k \leftarrow [1, N]$
   (c) sets $\hat{G} = \mathsf{Encoding}(\mathcal{H}(s_k, \beta))$
   (d) randomly selects $x \leftarrow [1, q-1]$
   (e) sends β and $X = x \cdot \hat{G}$ to $\mathcal{A}$
4. $\mathcal{A}$ outputs $q_\ell + 1$ points $Y, Z_1, \ldots, Z_t \in \mathcal{E}[q]$

$\mathcal{A}$ is said to succeed in security game G1 if $Y \neq \mathcal{O}$ and there exists $i \in [1, q_\ell]$ such that $Z_i = xY$. We then say that $\mathcal{A}$ $(\tau, N, q_\ell, \varepsilon)$-breaks G1 if $\mathcal{A}$ runs in at most τ elementary steps and $\mathcal{A}$ succeeds with probability at least $1/N + \varepsilon$.

To show that PACE v2 IM is AKE secure in the BPR model, we have to show that G1 cannot be solved with non-negligible ε. We first show that an adversary against G1 can be turned into an adversary against a new game G2 with a very close success probability. Here the security game G2 abstracts away the hash function $\mathcal{H}$ and is obtained from G1 by replacing $\mathcal{H}$ with an ideal hash function; in the random oracle model, $\mathcal{H}$ is viewed as a randomly selected function that maps $(\ell_1 + \ell_2)$-bit strings to elements of $[0, 2^{|p|-1}[$. We reformulate the security game G1 as the following game G2:

Security Game G2:

1. $\mathcal{C}$ generates public parameters and sends them to $\mathcal{A}$
2. $\mathcal{C}$ randomly selects $u_1, \ldots, u_N \leftarrow \{0,1\}^{|p|-1}$ and sends them to $\mathcal{A}$
3. $\mathcal{C}$:
   (a) randomly selects $k \leftarrow [1, N]$
   (b) sets $\hat{G} = \mathsf{Encoding}(u_k)$
   (c) randomly selects $x \leftarrow [1, q-1]$
   (d) sends $X = x \cdot \hat{G}$ to $\mathcal{A}$
4. $\mathcal{A}$ outputs $q_\ell + 1$ points $Y, Z_1, \ldots, Z_t \in \mathcal{E}[q]$

We say that $\mathcal{A}$ succeeds if $Y \neq \mathcal{O}$ and there exists $i \in [1, q_\ell]$ such that $Z_i = xY$. $\mathcal{A}$ is said to $(\tau, N, q_\ell, \varepsilon)$-break G2 if $\mathcal{A}$ runs in at most τ elementary steps and succeeds with probability at least $1/N + \varepsilon$.

Lemma 3. *In the random oracle model, an adversary who $(\tau, N, q_\ell, \varepsilon)$-breaks G1 can be turned into an adversary that $(\tau', N, q_\ell, \varepsilon')$-breaks G2, with $\varepsilon' = \varepsilon - q_h 2^{-\ell_2}$ and $\tau' \approx \tau$, where q_h is the number of hash queries in G1.*

Proof. We construct a reduction algorithm which behaves as per game G2 with a challenger and plays the role of the challenger in the G1 game towards an adversary that $(\tau, N, q_\ell, \varepsilon)$-breaks G1. Our reduction simulates the hash function $\mathcal{H}$ with random values except that $\mathcal{H}(s_n, \beta)$ is defined as u_n for $n \in [1, N]$ where $s_1, \ldots, s_N$ where chosen by the adversary and β was chosen at random by the reduction. The reduction simply forwards all the other elements (parameters, X, Y and all the points $Z_1, \ldots, Z_t$). It is easily seen that the simulation of $\mathcal{H}$ is perfect unless the adversary against G1 queries $\mathcal{H}$ on input (s_n, β) for some $n \in [1, N]$ prior to the random choice of β; this happens with probability at most $q_{\mathcal{H}} 2^{-\ell_2}$ where $q_{\mathcal{H}}$ is the total number of calls to $\mathcal{H}$. Therefore our reduction succeeds in $(\tau', N, q_\ell, \varepsilon')$-breaking G2 where $\varepsilon' = \varepsilon - q_h 2^{-\ell_2}$ and $\tau' \approx \tau$. □

Lemma 4 (Security of G2 under GCBDH). *Assume that GCBDH cannot be $(q_{\mathrm{DDH}}, \tau', \varepsilon')$-broken. Then there is no $(\tau, N, q_\ell, \varepsilon)$-adversary against game G2, where $q_{\mathrm{DDH}} = N \cdot q_\ell$,*

$$\varepsilon = 2\varepsilon' + 2 \cdot N 2^{-\lambda}$$

and $\tau' \simeq \tau$, where λ is a security parameter.

Proof. The proof is similar to the proof of Theorem 1 when considering the online phase in Game 5. Let us denote $\mathsf{Encoding}(u_n)$ by $\hat{G}_n$ for $n \in [1, N]$ and let x_n be the discrete logarithm of the point X (chosen by the challenger) in base $\hat{G}_n$. We consider the event $\mathsf{Ev}[\mathsf{double}]$ that there exists a pair of indices $i, j \in [1, q_\ell], i \neq j$ and a pair of indices $n, m \in [1, N], n \neq m$ such that $Z_i = x_n Y$ and $Z_j = x_m Y$. Clearly, if $\mathsf{Ev}[\mathsf{double}]$ never occurs during the game, then there can exist at most one pair $(i, n) \in [1, q_\ell] \times [1, N]$ such that $Z_i = x_n Y$ and by a standard information-theoretic argument, it follows that the success probability of the adversary is at most $1/N$. Assuming that the adversary succeeds with probability at least $1/N + \varepsilon$, we then get that $\Pr[\mathsf{Ev}[\mathsf{double}]] \geq \varepsilon$.

We now construct a reduction algorithm which $(q_{\mathrm{DDH}}, \tau', \varepsilon')$-breaks the GCBDH problem given an adversary that $(\tau, N, q_\ell, \varepsilon)$-breaks G2 with $q_{\mathrm{DDH}} = N \cdot q_\ell$. We receive as input a random tuple $(G, aG, bG) \leftarrow \mathcal{E}[q]^3$. For every $n \in [1, N]$, we run the simulator $\mathcal{S}$ from Lemma 2 either on the point aG to get a random pair (u_n, ρ_n) such that $\mathsf{Encoding}(u_n) = \rho_n(aG)$, or on the point bG to get a random pair (u_n, ρ_n) such that $\mathsf{Encoding}(u_n) = \rho_n(bG)$, with equal probability. Therefore we have $\hat{G}_n = \rho_n \xi_n G$ where $\xi_n \in \{a, b\}$.

This results in the generation of values $u_1, \ldots, u_N$ that are $2^{-\lambda}$-statistically close to the uniform distribution in $[0, 2^{|p|-1}[$. The success probability of $\mathcal{A}$ must then be at least $\varepsilon - N2^{-\lambda}$.

When $\mathcal{A}$ returns $q_\ell + 1$ points $Y, Z_1, \ldots, Z_t$, $Y \neq \mathcal{O}$, by using the DDH oracle our reduction algorithm checks whether $\mathrm{DDH}(\hat{G}_n, X, Y, Z_i) = 1$ for $n \in [1, N]$ and $i \in [1, q_\ell]$. From the observation above, this leads with probability at least ε to two pairs $(i, n), (j, m)$ such that $\mathrm{DDH}(\hat{G}_n, X, Y, Z_i) = 1$ and $\mathrm{DDH}(\hat{G}_m, X, Y, Z_j) = 1$. Since we have $X = xG = x_n \hat{G}_n = x_n \rho_n \xi_n G$ and similarly $x = xG = x_m \hat{G}_m = x_m \rho_m \xi_m G$, this implicitly defines the equations

$$Z_i = x_n Y = \frac{x}{\rho_n \xi_n} Y \qquad \text{and} \qquad Z_j = x_m Y = \frac{x}{\rho_m \xi_m} Y$$

where $\xi_n, \xi_m \in \{a, b\}$. Since the mapping $\xi_n \mapsto \{a, b\}$ is chosen at random and is independent from the view of $\mathcal{A}$, we get that $\xi_n \neq \xi_m$ with probability $1/2$. Let us assume wlog that $\xi_n = a$ and $\xi_m = b$. Then our reduction stops and outputs

$$\left(Y, \frac{\rho_n}{x} Z_i, \frac{\rho_m}{x} Z_j\right) = \left(Y, \frac{1}{a} Y, \frac{1}{b} Y\right)$$

thereby succeeding in breaking GCBDH in no more than $q_{\mathrm{DDH}} = N q_\ell$ calls to the DDH oracle. Overall, our reduction succeeds with probability $\varepsilon' = \varepsilon/2 - N \cdot 2^{-\lambda}$. □

Finally, combining Theorem 2 and Lemma 3 and 4, we obtain that PACE v2 IM is secure in the BPR model:

Theorem 3 (PACE v2 IM AKE security). *Assume that the GCBDH problem is (t, τ, ε)-hard. In the random oracle model and in the ideal cipher model, we have:*

$$\mathsf{Adv}^{ake}_{PACEv2}(t, Q) \leq \frac{q_e}{N} + 2 \cdot q_e \cdot \varepsilon + q_e \cdot \varepsilon_{\mathrm{forge}} + \frac{2q_e N^2 + 9q_e^2 N + q_c q_e}{\min\{q, 2^{\ell_0}\}} + 2q_e N 2^{-k} + \frac{q_e q_h}{2^{\ell_2}}$$

where the password is chosen from a dictionary of size N, $t^* = t + \mathcal{O}(kq_e^2 + kq_h^2 + kq_c^2 + k^2)$ and $Q = (q_e, q_c, q_h)$, where k is a security parameter, q_e is the number of protocol executions launched by the adversary, q_h and q_e indicate the number of hash and cipher queries, and $\varepsilon_{\mathrm{forge}}$ is the probability of forging a MAC in less than $2q_e$ adaptive MAC queries and time less than t^*.

5.3 A note on untraceability and unlinkability of transactions

As noted by Fischlin *et al.* in [7], the proven AKE security of PACE v2 IM has two interesting side effects in the sense that additional properties, untraceability and unlinkability are realized to a certain extent.

Untraceability. This property means that given a recorded protocol transcript and a password π, it is practically unfeasible to tell whether the protocol was executed with respect to this password. This property readily extends to lists of protocol transcripts, thus making it impossible, if one is given a database of eavesdropped protocol executions, to tell whether some of those were executed by a given user even if his credentials are fully known. Untraceability provides a form of user privacy and can be a desired security property in some contexts. Note however that transactions are untraceable as long as they share the same cryptographic suite of primitives, but that this is no longer the case if several suites are used. Typically, different countries may have different elliptic curve parameters and since those are easily extracted from protocol transcripts, transcripts will reveal the nationality of users involved. To remedy this (should it be required), a common set of cryptographic parameters must be adopted by all countries.

Unlinkability. This second property states that given two transcripts, it is practically unfeasible to tell whether they were executed by the same user (whose password is unknown). In a wider scope, this means that one cannot identify common users across databases of eavesdropped transactions. Unlinkability also provides some form of user privacy but in a stronger sense than untraceability (it can be shown that one who can link transactions can also trace users in transactions). In this case as well, the same remark holds about the cryptographic parameters; however transactions executed by users making use of the same set of parameters cannot be linked efficiently by anyone.

6 Conclusion

This document describes the password-authenticated secure channel protocol PACE v2 Integrated Mapping and provides evidence that the security requirements desired for a wide adoption as an industry standard are realized. PACE v2 Integrated Mapping enjoys provable security while ensuring optimal performances on both embedded and non embedded architectures.

References

1. Patent Statement and Licensing Declaration Form for ITU-T/ITU-R Recommendation ISO/IEC Deliverable. Letter from Sagem Sécurité to ICAO New Technologies Working Group Iternational Civil Aviation Organization, Paris, May 04^{th} 2010.
2. Michel Abdalla, Emmanuel Bresson, and David Pointcheval. Password-based authenticated key exchange in the three-party setting. In *Public Key Conference, PKC 2005*, volume 3386 of *Lecture Notes in Computer Science*, pages 65–84. Springer, 2005.
3. Michel Abdalla and David Pointcheval. Simple password-based encrypted key exchange protocols. In *CT-RSA*, pages 191–208, 2005.
4. Mihir Bellare, David Pointcheval, and Phillip Rogaway. Authenticated key exchange secure against disctionary attacks. In *Eurocrypt 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 139–155. Springer, 2000.
5. S. M. Bellovin and M. Merritt. Encrypted Key Exchange: Password-Based Protocols Secure Against Dictionary Attacks. *Proceedings of the I.E.E.E. Symposium on Research in Security and Privacy, Oakland*, 1992.
6. S. M. Bellovin and M. Merritt. Augmented Encrypted Key Exchange: A Password-Based Protocol Secure Against Dictionary Attacks and Password File Compromise. *Proceedings of the 1st ACM Conference on Computer and Communications Security, ACM Press*, 1993.
7. Jens Bender, Marc Fischlin, and Dennis Kuegler. Security analysis of the pace key-agreement protocol. Cryptology ePrint Archive, Report 2009/624, 2009. `http://eprint.iacr.org/`.
8. Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In Joe Kilian, editor, *Crypto 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer, 2001.
9. Eric Brier, Jean-Sebastien Coron, Thomas Icart, David Madore, Hugues Randriam, and Mehdi Tibouchi. Efficient indifferentiable hashing into ordinary elliptic curves. `http://eprint.iacr.org/`.
10. Federal Office for Information Security (BSI). Advanced security mechanism for Machine Readable Travle Documents – Extended Access Control (EAC), Password Authenticated Connection Establishment (PACE), and Restricted Identification (RI). *BSI-TR-03110, Version 2.0*, 2008.

11. ISO/IEC JTC1 SC17 WG3/TF5 for the International Civil Aviation Organization. Supplemental Access Control for Machine Readable Travel Documents. Technical Report, November 11, 2010.
12. Thomas Icart. How to Hash into Elliptic Curves (to appear). In *Crypto 2009*, Lecture Notes in Computer Science. Springer, 2009.
13. David Jablon. Cryptographic methods for remote authentication. *Patent Number 6226383, Filed by Integrity Sciences, Inc.*, 2001.
14. Jonathan Katz, Rafail Ostrovsky, and Moti Yung. Efficient password-authenticated key exchange using human-memorable passwords. In *EUROCRYPT*, pages 475–494, 2001.
15. Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. Cryptology ePrint Archive, Report 2004/332, 2004. `http://eprint.iacr.org/`.

A Proof of Lemma 2

The proof is base on the following Definitions, Lemma and Theorem from [9].

Definition 6 (Generalized Admissible Encoding [9], Def. 6). *A function $F : S \to R$ is said to be an ε-generalized admissible encoding if it satisfies the following properties:*

1. *Computable: F is computable in deterministic polynomial time;*
2. *Invertible: there exists a probabilistic polynomial time algorithm $\mathcal{I}_F$ such that $\mathcal{I}_F(r) \in F^{-1}(r) \cup \{\perp\}$ for all $r \in R$, and the distribution of $\mathcal{I}_F(r)$ is ε-statistically indistinguishable from the uniform distribution in S when r is uniformly distributed in R.*

F is an generalized admissible encoding if ε is a negligible function of the security parameter.

Definition 7 (Weak Encoding [9], Def. 5). *A function $f : S \to R$ between finite sets is said to be an α-weak encoding if it satisfies the following properties:*

1. *Computable: f is computable in deterministic polynomial time.*
2. *α-bounded: for s uniformly distributed in S, the distribution of $f(s)$ is α-bounded in R,* i.e. *the inequality $\Pr_s[f(s) = r] \leq \alpha/\#R$ holds for any $r \in R$.*
3. *Samplable: there is an efficient randomized algorithm $\mathcal{I}$ such that $\mathcal{I}(r)$ induces the uniform distribution in $f^{-1}(r)$ for any $r \in R$. Additionally $\mathcal{I}(r)$ returns $N_r = \#f^{-1}(r)$ for all $r \in R$.*

The function f is a weak encoding *if α is a polynomial function of the security parameter.*

Lemma 5 (Icart's function [9], Lem. 4). *Icart's function $f_{a,b}$ is an α-weak encoding from $\mathbb{F}_p$ to $E_{a,b}(\mathbb{F}_p)$, with $\alpha = 4N/p$, where N is the order of $E_{a,b}(\mathbb{F}_p)$.*

Lemma 6 (Simplified SWU algorithm [9], Lem. 6). *The simplified SWU algorithm has pre-image size at most 8 and can be inverted on its image in polynomial time. Then $f'_{a,b}$ is an α-weak encoding with $\alpha = 8N/q$, where N is the elliptic curve order.*

Theorem 4 (Weak $\to$ Generalized Admissible Encoding [9], Th. 3). *Let $\mathbb{G}$ be cyclic group of order N noted additively, and let G be a generator of $\mathbb{G}$. Let $f : S \to \mathbb{G}$ be an α-weak encoding. Then the function $F : S \times \mathbb{Z}_N \to \mathbb{G}$ with $F(s, x) := f(s) + xG$ is an ε-admissible encoding into $\mathbb{G}$, with $\varepsilon = (1 - 1/\alpha)^t$ for any t polynomial in the security parameter k, and $\varepsilon = 2^{-k}$ for $t = \alpha \cdot k$.*

A.1 Proof of Lemma 2

The proof of Lemma 2 is a straightforward consequence of Lemma 5, Lemma 6 and Theorem 4. Namely from Lemma 5 and Theorem 4 the function $F : \mathbb{F}_p \times \mathbb{F}_q \mapsto \mathcal{E}[q]$ defined as:

$$F(s, x) = \mathsf{Encoding}(s) + xG$$

is a generalized admissible encoding as per Definition 6, when $\mathsf{Encoding}$ is Icart's function. The same holds for the simplified SWU algorithm thanks to Lemma 6.

Therefore for both Icart and simplified SWU there exists a probabilistic polynomial-time inverting algorithm $\mathcal{I}$ that given a point $P \in \mathcal{E}[q]$ outputs $(s, x) \in (\mathbb{F}_p \cup \{\perp\}) \times \mathbb{F}_q$ such that

$$P = \mathsf{Encoding}(s) + xG$$

and the distribution of (s, x) is 2^{-k}-statistically indistinguishable from the uniform distribution in $(\mathbb{F}_p, \mathbb{F}_q)$.

Therefore given a point G as input, we generate a random $\beta \in \mathbb{Z}_q$ and give the point $P = \beta G$ as input to the inverting algorithm $\mathcal{I}$. The inverting algorithm returns (s, x) such that $P = f(s) + xG$. This gives $f(s) = P - xG = (\beta - x)G = \rho G$ where $\rho := \beta - x$ as required; moreover since by construction P is uniformly distributed in $\mathcal{E}[q]$ the distribution of s is 2^{-k}-statistically indistinguishable from the uniform distribution.